THE TRANSLATION OF DATA STRUCTURE REPRESENTATIONS OF SIMPLE QUEUING PROBLEMS INTO GPSS PROGRAMS AND ENGLISH TEXT

Robert Thomas McGee

United States
Naval Postgraduate School

THESIS

The Translation of Data Structure
Representations of Simple Queuing Problems
into GPSS Programs and English Text

by

Robert Thomas McGee

Thesis Advisor: George E. Heidorn

June 1971

Approved for public release; distribution unlimited.

The Translation of Data Structure Representations of Simple Queuing Problems into GPSS Programs and English Text

by

Robert Thomas McGee
Lieutenant, United States Navy
B.S., Brown University, 1964

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1971

ABSTRACT

One of the goals of computer technology is to have the ability to communicate with the computer in a natural language such as English. A research effort underway at the Naval Postgraduate School involves the design and implementation of a computer system for translating natural language descriptions of simulation problems into executable computer programs. In this system, English text is translated into an internal data structure which is then translated into a computer program for performing the simulation.

This thesis reports on an effort made to aid the user of this system by (1) extending the capabilities of an existing procedure for translating the internal data structure into a GPSS simulation program, and (2) developing a procedure for translating the data structure into English text so the user could see that his input text had been correctly interpreted. The basic operation of the system is described and examples are given to illustrate the system's capabilities.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

One of the goals of computer technology is to have the ability to communicate with the computer in a natural language such as English. Ideally, the user would not need to know anything about computer hardware or software or programming languages. He would explain the work to be done or ask questions of the computer and supply input data, all in a natural language. The computer would then perform the necessary work and supply the answer in a natural language format. To date, little of this type of work with natural language input to a computer has produced anything of practical value.

In early attempts to develop a natural language processor, brute force techniques were tried and, for the most part, discarded as both unwieldy and unworkable. Current attempts revolve around the development of language theories. Probably the most widely known natural language theory is that of transformational grammar by Noam Chomsky [1]. Another is that of stratificational grammar by Sydney Lamb [2-4]. A summary of current efforts in natural language computer processing may be found in Refs. 5 and 6.

A natural language processor (NLP) currently being developed at the Naval Postgraduate School [7] uses Lamb's theory of a stratified grammar in processing natural language. The immediate goal of NLP is the translation of a natural language expression of a simulation problem into a computer program written in a simulation programming language. By initially limiting the scope of the natural language input to a subset of English, an attempt is being made to produce a system of some immediate practical value, a system which can be used by persons who wish to solve simulation problems without learning the intricacies of a computer

simulation language. The long range goal of NLP is the translation of any input language to any output language. The method of processing involves the decoding of the input language text into an internal data structure (IDS) and then the encoding of the IDS into an output language text, all under the control of a FORTRAN program, guided by appropriate sets of decoding and encoding "rules."

The objective of the research being reported on in this thesis was to aid the user of NLP by (1) improving the appearance of the output simulation program, currently in GPSS, and (2) making it possible to translate the IDS into English so the user could see if his problem had been stated correctly. The first area involved expanding on and testing work done by LCDR Richard Hansen [8] in translating the IDS into GPSS programs. The second area involved developing a system to translate the IDS into an English language text.

This thesis begins by presenting a brief explanation of NLP as it relates to the achievement of the above objectives. The following two sections are then devoted to a description of the systems developed, and the final section presents conclusions and recommendations for future work. Reference 8 contains a detailed discussion of NLP and the development of the IDS-GPSS encoding system. Familiarity with the material presented there is necessary for a thorough understanding of this thesis.

II. THE OPERATION OF NLP

In order to introduce the reader to the procedure for translating the IDS to GPSS, English or any other language, this section will present the overall operations of NLP as they apply to this thesis. Discussion of the natural-language-to-IDS operation of NLP will be limited to that necessary to understand the development of the IDS.

A. BASIC OPERATION

The basic operation of NLP is extremely straightforward; the input language is decoded into the IDS, and then the IDS is encoded into the output language. The main idea behind the operation of NLP, and therefore the underlying reason for the form of the IDS, is that of extracting the meaning from the input language. The IDS is structured to store those elements which "contain" the meaning. A vital point in viewing the translation process is that the input text, the IDS, and the output text are merely three roughly equivalent representations of one meaning. An example may clarify the concept of extracting the meaning. Under the decoding process, the sentences "John hit Mary" and "Mary was hit by John" would result in exactly the same IDS:

sup -- hit
agent -- John
goal -- Mary

The meaning is then available in the IDS for encoding into the desired output language.

B. THE STRUCTURE OF IDS

The major conceptual building block of the IDS is the "record." A record represents an "entity," where the entity may be a complex sentence or a simple noun. The flexible structure of the record allows it to expand and contract as necessary to contain the distinguishing attribute values of the entity. A simplified record for "Customers arrive at the bank every 10 minutes," would have the following attribute-value pairs:

Attribute	Value
SUP (entity type)	arrive
AGENT	customers
GOAL	
LOCATION	bank
IETM (inter-event time)	10 minutes

The actual form of this record in the IDS would be considerably more cryptic than is indicated above in order to conserve space in the IDS. Special types of attributes called indicators are employed by NLP in situations where the value of the attribute may be represented by a zero or one, and attributes such as "goal" would actually be a number in the IDS. A more detailed description of the computer representation of an IDS record is given in Ref. 8.

A special type of record which will be referred to throughout the thesis is the SEGMENT. A SEGMENT is a record which represents the information in that part of the IDS currently being processed. As a record, a SEGMENT may range from simple to extremely complex and may possess any number of attributes. A SEGMENT may represent a portion of an IDS entity or an entire entity. SEGMENTS may be of different types depending upon the information they represent. For example, a record representing an

IDS action would be called an ACT type SEGMENT. For each type of SEGMENT in the system there is a record called a SEGMENT TYPE record which contains information about that type of SEGMENT. For instance, a SEGMENT type record has an attribute which points to a character string which is the name of the SEGMENT TYPE (e.g. ACT). Another attribute possessed by a SEGMENT TYPE record is a list of encoding rules which begin with SEGMENTS of this type.

C. NLP ENCODING

The process of translation from the IDS to the target output language is called encoding. Encoding consists of two phases. The first phase involves the preprocessing of the encoding rules in a compilation-like step. In the second phase the IDS is used as input data to the processed rules in an execution-like step. The structure generated in the first phase is similar in many respects to the object module produced as a result of a FORTRAN compilation. The structure may be saved and used to translate as many different IDS's as desired. Although the appearance of encoding rules for different target languages may vary considerably, the format is identical for them all. Appendices B and D are listings of the sets of rules developed in this research.

D. NLP ENCODING RULE FORMAT

All encoding rules follow the format:

```
SEGMENT TYPE (condition 1, condition 2, ....)    -->  ...
SEGMENT TYPE (action 1, action 2, ....)          ...
SEGMENT TYPE (action 1, action 2, ....)          ...
.
.
.
.
```

For a particular IDS SEGMENT, rule scan begins at the head of the list of rules for that SEGMENT TYPE and continues until the conditions existing in that SEGMENT match the conditions of a rule on the list, or until the default is taken due to failure to find any applicable rule. The default option simply causes the EBCDIC string in the ANMS attribute of the SUP attribute of the SEGMENT to be output. Once one applicable rule is found, the remaining rules are ignored. The conditions may range from specifying only that an attribute be present in the SEGMENT, to requiring that the attribute possess a certain value. A rule may have several conditions or none at all (in which case there would exist only one rule of that SEGMENT TYPE).

After an applicable rule is found, new SEGMENTS are created as dictated by the portion of the rule following the conversion symbol (--\>). The newly created SEGMENT(S) may have the same SEGMENT TYPE as the old SEGMENT or a new SEGMENT TYPE. In either situation the new SEGMENT(S) may possess any or all of the attributes of the old SEGMENT and may, in addition, have new attributes with specified values. Appendix A of Ref. 8 gives a complete BNF description of the encoding rules, and Appendix B of Ref. 8 contains the encoding rule symbology and further explains the encoding rule format.

E. THE IDS ENCODING PROCESS

Once the encoding rules have been "compiled," the "execution" phase may begin. The processing of the IDS is accomplished through the operation of a dual push-down stack. One side of the stack contains SP's, pointers to SEGMENT records, and the other side contains STP's, pointers to SEGMENT TYPE records. The basic cycle of stack operation begins by popping off

the top SP and STP. The SEGMENT TYPE record is obtained through the STP and is examined. If it is a terminal SEGMENT TYPE record, then the value of the ANMS attribute, e.g. "A," of the SEGMENT TYPE is written out. If the SEGMENT TYPE record is not a terminal SEGMENT TYPE record, the ANMS attribute is examined to see if it has the value "OUTPUT." If it does, the SEGMENT pointed to by the SP is accessed. The action then taken depends upon the attributes of the "OUTPUT" SEGMENT. The possible actions are:

1. Skipping lines in the written output
2. Shifting the printer to a desired column
3. Printing an EBCDIC string
4. Printing an integer number
5. Printing a decimal number

If the "OUTPUT" SEGMENT has no attributes, no output operation is performed.

Upon failure of the first two examinations, the list of rules pointed to by an attribute of the SEGMENT TYPE record is obtained, and the list is scanned, rule by rule. The SEGMENT pointed to by the SP is tested to see if it satisfies the conditions of the rule currently being considered. If the SEGMENT does not completely satisfy any of the rules, then the default option is taken; that is, the record pointed to by the SUP attribute of the SEGMENT is accessed and the EBCDIC string pointed to by the ANMS attribute is written out. If a rule is satisfied, then rule scan stops and the actions specified by the right part of the rule are taken. SP's and STP's for newly created SEGMENTS are placed on the stack in the inverse order of their creation; that is, bottom first and top last. Finally, the cycle is completed by erasing the SP and the STP which were popped off and erasing the SEGMENT pointed to by the SP.

To begin the processing of the IDS, an initial STP is placed on the stack with a null SP. The initial SEGMENT TYPE always has one rule with no conditions, and therefore it is always satisfied. The basis cycle of stack operation is repeated until the stack is empty, at which time processing of the IDS is complete. Since SEGMENTS created during processing contain only copies of portions of the IDS or pointers to IDS records, the erasure of SEGMENTS does not affect the IDS. Records in the IDS may be altered by accessing them through the MEMORY record. The MEMORY record is a unique record which contains pointers to the important records in the IDS and other attributes which are used for counting and storing numbers. Any reference in the Encoding rules to either MEM or MEMORY provides direct access to this record. A complete graphical presentation of the operation of the dual push-down stack and the resulting output is illustrated in Figures 13 through 17 of Ref. 8.

III. EXTENSION OF GPSS ENCODING SYSTEM

This section deals with the extensions and modifications to GES: A Data-Structure-to-GPSS Encoding System developed by Hansen [8]. The primary goal of this portion of the work was to make GES more useful to the user with little background in GPSS, and therefore the emphasis was in improving the appearance and readability of the GPSS program and its resulting output. As the work progressed, several other changes were made to GES in the interest of extending the system capability and increasing the visibility of the flow in the rule processing. However, all of these changes were just modifications to or extensions of the already sound structure of GES, and therefore the resulting system has been labeled XGES: An Extended GPSS Encoding System.

A. IMPROVED OUTPUT APPEARANCE

The solution to the somewhat cryptic appearance of the GPSS program and its output was a two-step process. Investigation of GPSS documentation [9] revealed that the addition of EQU cards to the GPSS program could increase the readability of both the GPSS program and the program's output. By simply equating an entity name, e.g. ship, and its internal identification number, the GPSS assembler would replace all occurrences of the entity identification number with the entity name in both the GPSS program and GPSS output. The second step in the process involved adding the additional attribute, IDNAME, to the list of attributes of an entity. The encoding rules to produce the EQU card were written so that if the user provided an IDNAME in his description of the problem, then an EQU

card would be output. Otherwise, no EQU card would be produced and the program appearance would remain as it was produced by GES. Figure 1 is a description of a harbor facility queuing problem. Figures 2 through 4 illustrate the direct input version of the IDS, the GPSS program, and the output from the GPSS program when this additional feature is used. A direct comparison of Figures 1 and 2 may be made with Figures 19 and 20 of Ref. 8 for the same problem. A final minor change made to improve the appearance was to alter the GES encoding rules to output the normal and exponential function definitions only when required, rather than arbitrarily producing them for every problem.

EXAMPLE PROBLEM

There is a port containing a harbor, 3 docks, 2 piers, a depot, and a barge. Ships arrive at the port with an interarrival time of 5 hours, uniformly distributed, with a range of $\pm$ 1 hour. 50% of the ships are blue ships, 30% are red, and 20% are green. After a ship arrives at the port, it unloads cargo at any available dock. Each dock has a capacity of 1 unit. Each ship takes up 1 unit of capacity. Unloading time at the dock is normally distributed as follows:

blue ship - mean of 5 hours, std dev of 1.5 hours
red ship - mean of 4 hours, std dev of 1.0 hours
green ship - mean of 3 hours, std dev of .5 hours

After unloading at a dock, a blue ship unloads cargo at the barge, a red ship unloads cargo at the depot, and a green ship unloads cargo at a pier. The barge has a capacity of 1 unit, a pier has a capacity of 1 unit, the depot has a capacity of 4 units. Unloading times are as follows:

barge - 1.5 hours, exponentially distributed
depot - 1 hour, exponentially distributed
pier - 1 hour, normally distributed, std dev of 15 minutes

Next, after these latest unloadings, 40% of the ships load cargo at a dock, and the remainder wait in the harbor. Dock loading time is 2 hours for any ship. After loading cargo at a dock, a ship waits in the harbor. A ship waits in the harbor until the barge is unoccupied. After waiting in the harbor, a ship leaves the port. The basic time unit is the minute. Problem duration is 4 days.

Figure 1 - Description of Harbor Problem

```
'REC1'    ('ARRIV',IDNO=1,SUCC='REC2',AGENT='REC11',
           LOCATION='REC71',IETM='REC41',ASNDISTR='REC47')

'REC2'    ('UNLOAD',IDNO=2,PRED='REC1',SUCC='REC61',
           AGENT='REC11',GOAL='REC13',LOCATION='REC72',
           DURATION='REC42')

'REC3'    ('UNLOAD',IDNO=3,PRED='REC2',SUCC='REC62',
           AGENT='REC15',GOAL='REC13',LOCATION='REC74',
           DURATION='REC44')

'REC4'    ('UNLOAD',IDNO=4,PRED='REC2',SUCC='REC62',
           AGENT='REC17',GOAL='REC13',LOCATION='REC75',
           DURATION='REC45')

'REC5'    ('UNLOAD',IDNO=5,PRED='REC2',SUCC='REC62',
           AGENT='REC19',GOAL='REC13',LOCATION='REC76',
           DURATION='REC46')

'REC6'    ('LOAD',IDNO=6,PRED='REC96',SUCC='REC7',
           AGENT='REC11',GOAL='REC13',LOCATION='REC72',
           DURATION='REC210')

'REC7'    ('WAIT',IDNO=7,PRED='REC97',SUCC='REC8',
           AGENT='REC11',LOCATION='REC77',DURATION='REC81')

'REC8'    ('LEAV',IDNO=8,PRED='REC7',AGENT='REC11',
           LOCATION='REC71')


'REC11'   ('SHIP',IDNO=1,CONSUMP='REC212', IDNAME="SHIP",
                CLASATR='COLOR')

'REC12'   ('PORT',IDNO=2,QUANTITY=1,STORIND=1, CAPACITY=
           'REC221',IDNAME="PORT")

'REC13'   ('CARGO',IDNO=3, IDNAME="CARGO")

'REC14'   ('DOCK',IDNO=4,LOCATION='REC73',QUANTITY=3,
           CAPACITY='REC212',STORIND=1, IDNAME="DOCKS")

'REC15'   ('SHIP',IDNO=5,QUANTITY='REC205',COLOR='GREEN',
           STRUC='REC11', IDNAME="GSHIP")

'REC16'   ('PIER',IDNO=6,LOCATION='REC73',QUANTITY=2,
           CAPACITY='REC212',STORIND=1, IDNAME="PIERS")

'REC17'   ('SHIP',IDNO=7,QUANTITY='REC207',COLOR='RED',
           STRUC='REC11', IDNAME="RSHIP")

'REC18'   ('DEPOT',IDNO=8,LOCATION='REC73',QUANTITY=1,
           CAPACITY='REC220',STORIND=1, IDNAME="DEPOT")

'REC19'   ('SHIP',IDNO=9,QUANTITY='REC208',COLOR='BLUE',
           STRUC='REC11', IDNAME="BSHIP",INTWIDTH=50,
           NUMINT=22,LOWINT=200)

'REC20'   ('BARGE',IDNO=10,LOCATION='REC73',QUANTITY=1,
           CAPACITY='REC212', IDNAME="BARGE")

'REC21'   ('HARBOR',IDNO=11,LOCATION='REC73',QUANTITY=1,
           STORIND=1, CAPACITY='REC221', IDNAME="HRBR")
```

FIGURE 2 - IDS REPRESENTATION OF HARBOR PROBLEM

```
'REC41'   ('UNIFORM',MEAN='REC201',RANGE='REC202')

'REC42'   ('NORMAL',MEAN='REC43',STDEV='REC48')

'REC43'   ('TYPTABL',FNID=3,FNARG='REC203',DORC="D",
            XYLAST=106,@101='REC15',@102='REC213',
            @103='REC17',@104='REC214',@105='REC19',
            @106='REC201')

'REC44'   ('NORMAL',MEAN='REC202',STDEV='REC206')

'REC45'   ('EXPON',MEAN='REC202')

'REC46'   ('EXPON',MEAN='REC209')

'REC47'   ('TYPDIST',FNID=4,PNUM='REC212',FNARG='REC211',
            DORC="D",XYLAST=106,@101='REC215',@102='REC15',
            @103='REC216',@104='REC17',@105='REC217',
            @106='REC19')

'REC48'   ('TYPTABL',FNID=5,FNARG='REC203',DORC="D",
            XYLAST=106,@101='REC15',@102='REC218',
            @103='REC17',@104='REC202',@105='REC19',
            @106='REC209')

'REC61'   ('PTYP',SUCARG='REC203',XYLAST=106,@101='REC15',
            @102='REC3',@103='REC17',@104='REC4',
            @105='REC19',@106='REC5')

'REC62'   ('FRACTNL',SUCARG='REC211',XYLAST=104,
            @101='REC219',@102='REC6',@103='REC217',
            @104='REC7')

'REC71'   ('AT',LOCOBJ='REC12')

'REC72'   ('AT',LOCOBJ='REC14')

'REC73'   ('IN',LOCOBJ='REC12')

'REC74'   ('AT',LOCOBJ='REC16')

'REC75'   ('AT',LOCOBJ='REC18')

'REC76'   ('AT',LOCOBJ='REC20')

'REC77'   ('IN',LOCOBJ='REC21')

'REC81'   ('COND1',CONDENTY='REC20')

'REC91'   ('MOBLIST',LASTREC=14,@11='REC11',@12='REC15',
            @13='REC17',@14='REC19')

'REC92'   ('STALIST',LASTREC=17,@11='REC12',@12='REC13',
            @13='REC14',@14='REC16',@15='REC18',
            @16='REC20',@17='REC21')

'REC93'   ('ACTNLIST',LASTREC=18,@11='REC1',@12='REC2',
            @13='REC3',@14='REC4',@15='REC5',@16='REC6',
            @17='REC7',@18='REC8')
```

FIGURE 2 (CONTINUED)

```
'REC94'   ('DSTRLIST',LASTREC=18,@11='REC41',@12='REC42',
           @13='REC43',@14='REC44',@15='REC45',@16='REC46',
           @17='REC47',@18='REC48')

'REC95'   ('SCSRLIST',LASTREC=15,@11='REC2',@12='REC61',
           @13='REC62',@14='REC7',@15='REC8')

'REC96'   ('PREDLIST',LASTREC=13,@11='REC3',@12='REC4',
           @13='REC5')

'REC97'   ('PREDLIST',LASTREC=14,@11='REC3',@12='REC4',
           @13='REC5',@14='REC6')

'REC201'  ('MINUTE',NUM=300)

'REC202'  ('MINUTE',NUM=60)

'REC203'  ('PARAMNO',NUM=1)

'REC204'  ('MINUTE',NUM=5760)

'REC205'  ('PERCENT',NUM=20)

'REC206'  ('MINUTE',NUM=15)

'REC207'  ('PERCENT',NUM=30)

'REC208'  ('PERCENT',NUM=50)

'REC209'  ('MINUTE',NUM=90)

'REC210'  ('MINUTE',NUM=120)

'REC211'  ('RANDM')

'REC212'  ('UNIT',NUM=1)

'REC213'  ('MINUTE',NUM=180)

'REC214'  ('MINUTE',NUM=240)

'REC215'  ('DECIMAL',NUM=200)

'REC216'  ('DECIMAL',NUM=500)

'REC217'  ('DECIMAL',NUM=1000)

'REC218'  ('MINUTE',NUM=30)

'REC219'  ('DECIMAL',NUM=400)

'REC220'  ('UNIT',NUM=4)

'REC221'  ('UNIT', NUM=1000)

SETMEM    (RNNO(MEMORY)=1,MEPTR(MEMORY)='REC91',
           DISTPTR(MEMORY)='REC94',SUCPTR(MEMORY)='REC95',
           PROBTIME(MEMORY)='REC204',MFNID(MEMORY)=6,
           SEPTR(MEMORY)='REC92',ACPTR(MEMORY)='REC93',
           MVARID(MEMORY)=1, EXPUSED(MEM),NORMUSED(MEM))
```

FIGURE 2 (CONTINUED)

```
        SIMULATE
        RMULT       277,423,715,121,655,531,999,813
PORT    EQU         2,S,Q
2       STORAGE     1000
CARGO   EQU         3,F,Q
DOCKS   EQU         4,S,Q
4       STORAGE     3
PIERS   EQU         6,S,Q
6       STORAGE     2
DEPOT   EQU         8,S,Q
8       STORAGE     4
BARGE   EQU         10,F,Q
HRBR    EQU         11,S,Q
11      STORAGE     1000
GSHIP   EQU         5,T
5       TABLE       M1,0,1,2
RSHIP   EQU         7,T
7       TABLE       M1,0,1,2
BSHIP   EQU         9,T
9       TABLE       M1,200,50,22
1       FUNCTION    RN1,C24
0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915/.7,1.2/.75,1.39/
.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52/.94,2.81/.95,2.99/.96,3.2/
.97,3.5/.98,3.9/.99,4.6/.995,5.3/.998,6.2/.999,7/.9997,8/
2       FUNCTION    RN2,C29
0,-3/.012,-2.25/.027,-1.93/.043,-1.72/.062,-1.54/.084,-1.38/
.104,-1.26/.131,-1.12/.159,-1/.187,-.89/.23,-.74/.267,-.62/.334,-.43/
.432,-.17/.5,0/.568,.17/.666,.43/.732,.62/.77,.74/.813,.89/.841,1/
.869,1.12/.896,1.26/.916,1.38/.938,1.54/.957,1.72/.973,1.93/
.988,2.25/1,3/
3       FUNCTION    P1,D3
GSHIP,180/RSHIP,240/BSHIP,300/
4       FUNCTION    RN3,D3
.200,GSHIP/.500,RSHIP/1.000,BSHIP/
5       FUNCTION    P1,D3
GSHIP,30/RSHIP,60/BSHIP,90/
6       FUNCTION    P1,D3
GSHIP,ACT3/RSHIP,ACT4/BSHIP,ACT5/
7       FUNCTION    RN4,D2
.400,ACT6/1.000,ACT7/
        GENERATE    300,60
        ASSIGN      1,FN4
        ENTER       PORT
```

FIGURE 3 - GPSS REPRESENTATION OF HARBOR PROBLEM

```
ACT2    QUEUE       DOCKS
        ENTER       DOCKS
        DEPART      DOCKS
        ADVANCE     V1
1       FVARIABLE   FN3+(FN5*FN2)
        LEAVE       DOCKS
        TRANSFER    ,FN6
ACT3    QUEUE       PIERS
        ENTER       PIERS
        DEPART      PIERS
        ADVANCE     V2
2       FVARIABLE   60+(15*FN2)
        LEAVE       PIERS
        TRANSFER    ,FN7
ACT4    QUEUE       DEPOT
        ENTER       DEPOT
        DEPART      DEPOT
        ADVANCE     60,FN1
        LEAVE       DEPOT
        TRANSFER    ,FN7
ACT5    QUEUE       BARGE
        SEIZE       BARGE
        DEPART      BARGE
        ADVANCE     90,FN1
        RELEASE     BARGE
        TRANSFER    ,FN7
ACT6    QUEUE       DOCKS
        ENTER       DOCKS
        DEPART      DOCKS
        ADVANCE     120
        LEAVE       DOCKS
ACT7    QUEUE       HRBR
        ENTER       HRBR
        DEPART      HRBR
        GATE NU     BARGE
        LEAVE       HRBR
ACT8    TABULATE    P1
        LEAVE       PORT
        TERMINATE
        GENERATE    5760
        TERMINATE   1
        START       1
        END
```

FIGURE 3 (CONTINUED)

```
BLOCK                                                                                    CARD
NUMBER   *LOC    OPERATION  A,B,C,D,E,F,G                COMMENTS                      NUMBER
                  SIMULATE                                                                  1
                  RMULT      277,423,715,121,655,531,999,813                                2
          PORT    EQU        2,S,Q                                                          3
          2       STORAGE    1000                                                           4
          CARGO   EQU        3,F,Q                                                          5
          DOCK    EQU        4,S,Q                                                          6
          4       STORAGE    3                                                              7
          PIER    EQU        6,S,Q                                                          8
          6       STORAGE    2                                                              9
          DEPO    EQU        8,S,Q                                                         10
          8       STORAGE    4                                                             11
          BARG    EQU        10,F,Q                                                        12
          HRBR    EQU        11,S,Q                                                        13
          11      STORAGE    1000                                                          14
          GSHIP   EQU        5,T                                                           15
          5       TABLE      M1,0,1,2                                                      16
          RSHIP   EQU        7,T                                                           17
          7       TABLE      M1,0,1,2                                                      18
          BSHIP   EQU        9,T                                                           19
          9       TABLE      M1,200,50,22                                                  20
          1       FUNCTION   RN1,C24                                                       21
        0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915/.7,1.2/.75,1.39/              22
        .8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52/.94,2.81/.95,2.99/.96,3.2/              23
        .97,3.5/.98,3.9/.99,4.6/.995,5.3/.998,6.2/.999,7/.9997,8/                        24
          2       FUNCTION   RN2,C29                                                       25
        0,-3/.012,-2.25/.027,-1.93/.043,-1.72/.062,-1.54/.084,-1.38/                     26
        .104,-1.26/.131,-1.12/.159,-1/.187,-.89/.23,-.74/.267,-.62/.334,-.43/            27
        .432,-.17/.5,0/.568,.17/.666,.43/.732,.62/.77,.74/.813,.89/.841,1/              28
        .869,1.12/.896,1.26/.916,1.38/.938,1.54/.957,1.72/.973,1.93/                     29
        .988,2.25/1,3/                                                                   30
          3       FUNCTION   P1,D3                                                        31
        GSHIP,180/RSHIP,240/BSHIP,300/                                                   32
          4       FUNCTION   RN3,D3                                                       33
        .200,GSHIP/.500,RSHIP/1.000,BSHIP/                                               34
          5       FUNCTION   P1,D3                                                        35
        GSHIP,30/RSHIP,60/BSHIP,90/                                                      36
          6       FUNCTION   P1,D3                                                        37
        GSHIP,LBL3/RSHIP,LBL4/BSHIP,LBL5/                                                38
          7       FUNCTION   RN4,D2                                                       39
        .400,LBL6/1.000,LBL7/                                                            40
1                 GENERATE   300,60                                                      41
2                 ASSIGN     1,FN4                                                       42
3                 ENTER      PORT                                                        43
4         LBL2    QUEUE      DOCK                                                        44
5                 ENTER      DOCK                                                        45
6                 DEPART     DOCK                                                        46
7                 ADVANCE    V1                                                          47
          1       FVARIABLE  FN3+(FN5*FN2)                                               48
8                 LEAVE      DOCK                                                        49
9                 TRANSFER   ,FN6                                                        50
10        LBL3    QUEUE      PIER                                                        51
11                ENTER      PIER                                                        52
12                DEPART     PIER                                                        53
13                ADVANCE    V2                                                          54
          2       FVARIABLE  60+(15*FN2)                                                 55
```

Figure 4 - GPSS Output for Harbor Problem

```
14              LEAVE       PIER                                          56
15              TRANSFER    ,FN7                                          57
16        LBL4  QUEUE       DEPO                                          58
17              ENTER       DEPO                                          59
18              DEPART      DEPO                                          60
19              ADVANCE     60,FN1                                        61
20              LEAVE       DEPO                                          62
21              TRANSFER    ,FN7                                          63
22        LBL5  QUEUE       BARG                                          64
23              SEIZE       BARG                                          65
24              DEPART      BARG                                          66
25              ADVANCE     90,FN1                                        67
26              RELEASE     BARG                                          68
27              TRANSFER    ,FN7                                          69
28        LBL6  QUEUE       DOCK                                          70
29              ENTER       DOCK                                          71
30              DEPART      DOCK                                          72
31              ADVANCE     120                                           73
32              LEAVE       DOCK                                          74
33        LBL7  QUEUE       HRBR                                          75
34              ENTER       HRBR                                          76
35              DEPART      HRBR                                          77
36              GATE NU     BARG                                          78
37              LEAVE       HRBR                                          79
38        LBL8  TABULATE    P1                                            80
39              LEAVE       PORT                                          81
40              TERMINATE                                                 82
41              GENERATE    5760                                          83
42              TERMINATE   1                                             84
                START       1                                             85
                END                                                       86
```

BLOCK NUMBER	SYMBOL	REFERENCES BY CARD NUMBER
4	LBL2	
10	LBL3	38
16	LBL4	38
22	LBL5	38
28	LBL6	40
33	LBL7	40
38	LBL8	

FACILITY SYMBOLS AND CORRESPONDING NUMBERS

10	BARG
3	CARGO

STORAGE SYMBOLS AND CORRESPONDING NUMBERS

8	DEPO
4	DOCK
11	HRBR
6	PIER
2	PORT

QUEUE SYMBOLS AND CORRESPONDING NUMBERS

10	BARG
3	CARGO
8	DEPO
4	DOCK
11	HRBR
6	PIER
2	PORT

TABLE SYMBOLS AND CORRESPONDING NUMBERS

9	BSHIP
5	GSHIP
7	RSHIP

Figure 4 - Continued

```
                RMULT     277,423,715,121,655,531,999,813
2               STORAGE   1000
4               STORAGE   3
6               STORAGE   2
8               STORAGE   4
11              STORAGE   1000
5               TABLE     M1      0       1       2
7               TABLE     M1      0       1       2
9               TABLE     M1      200     50      22
1               FUNCTION  FN1     C24
0                  0                .1              .104            .2              .222
.3                 .355             .4              .509            .5              .69
.6                 .915             .7              1.2             .75             1.39
.8                 1.6              .84             1.83            .88             2.12
.9                 2.3              .92             2.52            .94             2.81
.95                2.99             .96             3.2             .97             3.5
.98                3.9              .99             4.6             .995            5.3
.998               6.2              .999            7               .9997           8
2               FUNCTION  FN2     C29
0                  -3               .012            -2.25           .027            -1.93
.043               -1.72            .062            -1.54           .084            -1.38
.104               -1.26            .131            -1.12           .159            -1
.187               -.89             .23             -.74            .267            -.62
.334               -.43             .432            -.17            .5              0
.568               .17              .666            .43             .732            .62
.77                .74              .813            .89             .841            1
.869               1.12             .896            1.26            .916            1.38
.938               1.54             .957            1.72            .973            1.93
.988               2.25             1               3
3               FUNCTION  M1      D3
5                  180              7               240             9               300
4               FUNCTION  FN3     D3
.200               5                .500            7               1.000           9
5               FUNCTION  P1      D3
5                  30               7               60              9               90
6               FUNCTION  P1      D3
5                  10               7               16              9               22
7               FUNCTION  RN4     D2
.400               20               1.000           33
1               GENERATE  200     60
2               ASSIGN    1       FN4
3               ENTER     4
4               QUEUE     4
5               ENTER     4
6               DEPART    4
7               ADVANCE   M1
1               FVARIABLE FN3+(FN5*FN2)
8               LEAVE     4
9               TRANSFER          FN6
10              QUEUE     6
11              ENTER     6
12              DEPART    6
13              ADVANCE   M2
2               FVARIABLE 60+(15*FN2)
14              LEAVE     6
15              TRANSFER          FN7
16              QUEUE     8
17              ENTER     8
18              DEPART    8
19              ADVANCE   60      FN1
20              LEAVE     8
21              TRANSFER          FN7
22              QUEUE     10
23              SEIZE     10
24              DEPART    10
25              ADVANCE   90      FN1
26              RELEASE   10
27              TRANSFER          FN7
28              QUEUE     4
29              ENTER     4
30              DEPART    4
31              ADVANCE   120
32              LEAVE     4
33              QUEUE     11
34              ENTER     11
35              DEPART    11
36              GATE NU   10
37              LEAVE     11
38              TABULATE  M1
39              LEAVE     2
40              TERMINATE
41              GENERATE  5760
42              TERMINATE 1
```

Figure 4 - Continued

START 1

RELATIVE CLOCK 5760 ABSOLUTE CLOCK 5760

BLOCK COUNTS

BLOCK CURRENT	TOTAL	BLOCK CURRENT	TOTAL	BLOCK CURRENT	TOTAL	BLOCK CURRENT	TOTAL	BLOCK CURRENT	TOTAL	BLOCK CURRENT	TOTAL
1 0	18	11 0	1	21 0	6	31 0	5	41 0	1		
2 0	18	12 0	1	22 0	10	32 0	5	42 0	1		
3 0	18	13 0	1	23 0	10	33 0	16				
4 0	18	14 0	1	24 0	10	34 0	16				
5 0	18	15 0	1	25 1	10	35 0	16				
6 0	18	16 0	6	26 0	9	36 0	16				
7 1	18	17 0	6	27 0	9	37 0	16				
8 0	17	18 0	6	28 0	5	38 0	16				
9 0	17	19 0	6	29 0	5	39 0	16				
10 0	1	20 0	6	30 0	5	40 0	16				

FACILITY	AVERAGE UTILIZATION	NUMBER ENTPIES	AVERAGE TIME/TRAN	SEIZING TRANS. NO.	PREEMPTING TRANS. NO.
BARG	.137	10	79.399	4	

STORAGE	CAPACITY	AVERAGE CONTENTS	AVERAGE UTILIZATION	ENTRIES	AVERAGE TIME/TRAN	CURRENT CONTENTS	MAXIMUM CONTENTS
PORT	1000	1.118	.001	18	358.055	2	3
DOCK	3	.903	.301	23	226.347	1	2
PIER	2	.010	.005	1	61.000		1
DEPO	4	.061	.015	6	59.500		1
HRBR	1000	.004	.000	16	1.687		2

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/TRANS	$AVERAGE TIME/TRANS	TABLE NUMBER	CURRENT CONTENTS
DOCK	1	.000	23	23	100.0	.000	.000		
PIER	1	.000	1	1	100.0	.000	.000		
DEPO	1	.000	6	6	100.0	.000	.000		
BARG	1	.000	10	10	100.0	.000	.000		
HRBR	1	.000	16	16	100.0	.000	.000		

$AVERAGE TIME/TRANS = AVERAGE TIME/TRANS EXCLUDING ZERO ENTRIES

TABLE GSHIP

ENTRIES IN TABLE	MEAN ARGUMENT	STANDARD DEVIATION	SUM OF ARGUMENTS	
1	373.000		373.000	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PER CENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
0	0	.00	.0	100.0	-.000	-.000
OVERFLOW	1	100.00	100.0	.0		
AVERAGE VALUE OF OVERFLOW		373.00				

TABLE RSHIP

ENTRIES IN TABLE	MEAN ARGUMENT	STANDARD DEVIATION	SUM OF ARGUMENTS	
6	268.500	89.687	1611.000	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PER CENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
0	0	.00	.0	100.0	-.000	-2.993
OVERFLOW	6	100.00	100.0	.0		
AVERAGE VALUE OF OVERFLOW		268.50				

TABLE BSHIP

ENTRIES IN TABLE	MEAN ARGUMENT	STANDARD DEVIATION	SUM OF ARGUMENTS	
9	433.222	121.250	3899.000	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PER CENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
200	0	.00	.0	100.0	.461	-1.923
250	0	.00	.0	100.0	.577	-1.511
300	2	22.22	22.2	77.7	.692	-1.098
350	1	11.11	33.3	66.6	.807	-.686
400	1	11.11	44.4	55.5	.923	-.273
450	1	11.11	55.5	44.4	1.038	.138
500	2	22.22	77.7	22.2	1.154	.550
550	0	.00	77.7	22.2	1.269	.963
600	0	.00	77.7	22.2	1.384	1.375
650	2	22.22	100.0	.0	1.500	1.787

REMAINING FREQUENCIES ARE ALL ZERO

Figure 4 - Continued

B. ENTITY TRANSIT TIME

Entity transit time is simply the total time an entity spends in the system. In order to gather this statistic, GPSS requires a tabulate card to mark an entity's exit from the system (entry is marked automatically) and a table card for each different type of entity passing through the system. A simple modification to the rules produces the required cards. An interesting feature of the rule structure allows the user to provide the statistical parameters for the table definition cards or to default and let the system provide them. Figures 3 and 4 illustrate an example where the user has provided the parameter for one table definition card and lets the system provide the parameters for the others.

C. SHORTEST LINE CHOICE

A situation which arises repeatedly in simulation problems requires that an arriving customer make a choice of queues. Normally the action taken is to queue in the shortest line which provides the desired service. The GPSS equivalent is the select block. As no provisions for generation of this type of block exists under GES, rules were written to provide the capability in XGES. An example of a simulation problem which requires such a choice is the bank problem described in Figure 5. To demonstrate the flexibility of the encoding rules, two different IDS's were specified in direct format and are shown in Figures 6 and 7. The GPSS output from the XGES processing of the IDS's is shown in Figures 8 and 9. Although different in appearance, the two GPSS programs produce exactly the same results when executed.

EXAMPLE PROBLEM

Customers arrive at a bank in a truly random manner. One third of them have commercial accounts, one fourth of them have personal accounts, and the rest just have miscellaneous business. The bank has three windows for commercial accounts and two windows for personal accounts. When a customer with a commercial account arrives at the bank, he stands at the commercial accounts window which has the shortest line. When a customer with a personal account arrives, he stands at that personal accounts window which has the shortest line. Any other customer, upon arrival, stands at any window that happens to have the shortest line.

The customers are serviced at the windows, one at a time. The times to service customers are exponentially distributed, with means of 5, 4, and 2 minutes, for the commercial accounts, personal accounts, and the others, respectively. After being serviced, a customer leaves the bank.

If the mean interarrival time for customers at the bank is 10 minutes, what is the average length of time each type of customer is in the bank during a five-hour day, and what percent of the time are the windows busy?

Figure 5 - Description of Bank Problem

```
REC1      ('ARRIV', IDNO=1, SUCC='REC2', AGENT='REC11',
          LOCATION='REC81', IETM='REC45', ASNDISTR='REC51')

REC2      ('GOTO', IDNO=2, SUCC='REC41', PRED='REC1',
          ARGAZ='REC91',AGENT='REC11',LOCATION='REC84')

REC3      ('SERVIC', IDNO=3, PRED='REC2', SUCC='REC6',
          LOCATION='REC83', DURATION='REC42', GOAL='REC12')

REC4      ('SERVIC', IDNO=4, PRED='REC2', SUCC='REC6',
          LOCATION='REC83', DURATION='REC43', GOAL='REC13')

REC5      ('SERVIC', IDNO=5, PRED='REC2', SUCC='REC6',
          LOCATION='REC83', DURATION='REC44',GOAL='REC14')

REC6      ('LEAV', IDNO=6, PRED='REC92', AGENT='REC11',
          LOCATION='REC81')


REC11     ('CUSTOMER', IDNO=1, CONSUMP='REC61', IDNAME=
          "CUST1",CLASATR='TYPE')

REC12     ('CUSTOMER', IDNO=2, STRUC='REC11', IDNAME=
          "CUST2",TYPE='COMMERCE')

REC13     ('CUSTOMER', IDNO=3, STRUC='REC11', IDNAME=
          "CUST3",TYPE='PERSONAL')

REC14     ('CUSTOMER', IDNO=4, STRUC='REC11', IDNAME=
          "CUST4",TYPE='MISC')

REC15     ('WINDOW',IDNO=5,CAPACITY='REC61',CLASATR='TYPE',
          LOCATION='REC82', QUANTITY=5, IDNAME="WNDW1")

REC16     ('WINDOW', IDNO=6, STRUC='REC15', IDNAME=
          "WNDW2",TYPE='COMMMISC',QUANTITY=1)

REC17     ('WINDOW', IDNO=7, STRUC='REC15', IDNAME=
          "WNDW3",TYPE='COMMMISC',QUANTITY=1)

REC18     ('WINDOW', IDNO=8, STRUC='REC15', IDNAME=
          "WNDW4",TYPE='COMMMISC',QUANTITY=1)

REC19     ('WINDOW', IDNO=9, STRUC='REC15', IDNAME=
          "WNDW5",TYPE='PERMISC',QUANTITY=1)

REC20     ('WINDOW', IDNO=10,STRUC='REC15', IDNAME=
          "WNDW6",TYPE='PERMISC',QUANTITY=1)

REC21     ('BANK', IDNO=11, QUANTITY=1, STORIND=1, IDNAME=
          "BANK", CAPACITY='REC74')


REC41     ('PTYP', SUCARG='REC65', XYLAST=106, @101='REC12',
          @102='REC3',@103='REC13',@104='REC4',@105='REC14',
          @106='REC5')


REC42     ('EXPON', MEAN='REC67')

REC43     ('EXPON', MEAN='REC68')

REC44     ('EXPON', MEAN='REC69')
```

FIGURE 6 - IDS-1 REPRESENTATION OF BANK PROBLEM

```
REC45     ('EXPON', MEAN='REC72')

REC51     ('TYPDIST', FNID=3, PNUM='REC61', FNARG='REC66',
          DORC="D", XYLAST=106, @101='REC62', @102='REC12',
          @103='REC63',@104='REC13',@105='REC64',@106=
          'REC14')

REC52     ('TYPTABL', FNID=4, FNARG='REC65', DORC="D",
          XYLAST=106,@101='REC12',@102='REC16',@103='REC13',
          @104='REC19', @105='REC14', @106='REC16')

REC53     ('TYPTABL', FNID=5, FNARG='REC65', DORC="D",
          XYLAST=106,@101='REC12',@102='REC18',@103='REC13',
          @104='REC20', @105='REC14', @106='REC20')

REC61     ('UNIT', NUM=1)

REC62     ('DECIMAL', NUM=333)

REC63     ('DECIMAL', NUM=583)

REC64     ('DECIMAL', NUM=1000)

REC65     ('PARAMNO', NUM=1)

REC66     ('RANDM')

REC67     ('MINUTE', NUM=50)

REC68     ('MINUTE', NUM=40)

REC69     ('MINUTE', NUM=20)

REC70     ('UNIT', NUM=2)

REC71     ('PARAMNO', NUM=2)

REC72     ('MINUTE', NUM=10)

REC73     ('MINUTE', NUM=7200)

REC74     ('UNIT', NUM=100)

REC81     ('AT', LOCOBJ='REC21')

REC82     ('IN', LOCOBJ='REC21')

REC83     ('AT', LOCOBJ='REC71')

REC84     ('IN',LOCOBJ='REC71')

REC91     ('SELARGS', ARGA='REC71', ARGB='REC52',
          ARGC='REC53', ARGE="Q")

REC92     ('PREDLIST', LASTREC=13, @11='REC3', @12='REC4',
          @13='REC5')

REC93     ('MOBLIST', LASTREC=14, @11='REC11', @12='REC12',
          @13='REC13', @14='REC14')

REC94     ('STALIST', LASTREC=17, @11='REC15', @12='REC16',
          @13='REC17',@14='REC18',@15='REC19',@16='REC20',
          @17='REC21')
```

FIGURE 6 (CONTINUED)

```
REC95     ('ACTNLIST', LASTREC=16, @11='REC1', @12='REC2',
          @13='REC3', @14='REC4', @15='REC5', @16='REC6')

REC96     ('SCSRLIST', LASTREC=13, @11='REC2', @12='REC41',
          @13='REC6')

REC97     ('DSTRLIST', LASTREC=17, @11='REC42', @12='REC43',
          @13='REC44',@14='REC45',@15='REC51',@16='REC52',
          @17='REC53')

SETMEM    (RNNO(MEM)=1,MEPTR(MEM)='REC93',
          DISTPTR(MEM)='REC97',SUCPTR(MEM)='REC96',
          PROBTIME(MEM)='REC73',MENID(MEM)=6,
          SEPTR(MEM)='REC94',ACPTR(MEM)='REC95',MVARID(MEM)
          =1,EXPUSED(MEM))
```

FIGURE 6 (CONTINUED)

```
REC1      ('ARRIV', IDNO=1, SUCC='REC2', AGENT='REC11',
          LOCATION='REC81', IETM='REC45', ASNDISTR='REC51')

REC2      ('GOTO', IDNO=2, SUCC='REC3', PRED='REC1',
          ARGAZ='REC91',AGENT='REC11',LOCATION='REC15')

REC3      ('SERVIC', IDNO=3, PRED='REC2', SUCC='REC4',
          LOCATION='REC83', DURATION='REC44',GOAL='REC11')

REC4      ('LEAV', IDNO=4,PRED='REC3', AGENT='REC11',
          LOCATION='REC81')


REC11     ('CUSTOMER', IDNO=1, CONSUMP='REC61',
          IDNAME="CUST1",CLASATR='TYPE',LOWINT=5,INTWIDTH=
          3,NUMINT=30)

REC12     ('CUSTOMER', IDNO=2, STRUC='REC11', IDNAME=
          "CUST2",TYPE='COMMERCE')

REC13     ('CUSTOMER', IDNO=3, STRUC='REC11', IDNAME=
          "CUST3",TYPE='PERSONAL')

REC14     ('CUSTOMER', IDNO=4, STRUC='REC11', IDNAME=
          "CUST4",TYPE='MISC')

REC15     ('WINDOW',IDNO=5,CAPACITY='REC61',CLASATR='TYPE',
          LOCATION='REC82', QUANTITY=5, IDNAME="WNDW1")

REC16     ('WINDOW', IDNO=6, STRUC='REC15', IDNAME=
          "WNDW2",TYPE='COMMMISC',QUANTITY=1)

REC17     ('WINDOW', IDNO=7, STRUC='REC15', IDNAME=
          "WNDW3",TYPE='COMMMISC',QUANTITY=1)

REC18     ('WINDOW', IDNO=8, STRUC='REC15', IDNAME=
          "WNDW4",TYPE='COMMMISC',QUANTITY=1)

REC19     ('WINDOW', IDNO=9, STRUC='REC15', IDNAME=
          "WNDW5",TYPE='PERMISC',QUANTITY=1)

REC20     ('WINDOW', IDNO=10,STRUC='REC15', IDNAME=
          "WNDW6",TYPE='PERMISC',QUANTITY=1)

REC21     ('BANK', IDNO=11, QUANTITY=1, STORIND=1,
          CAPACITY='REC74', IDNAME="BANK")


REC44     ('EXPON',MEAN='REC54')

REC45     ('EXPON', MEAN='REC72')


REC51     ('TYPDIST', FNID=3, PNUM='REC61', FNARG='REC66',
          DORC="D", XYLAST=106, @101='REC62', @102='REC12',
          @103='REC63',@104='REC13',@105='REC64',@106=
          'REC14')

REC52     ('TYPTABL', FNID=4, FNARG='REC65', DORC="D",
          XYLAST=106,@101='REC12',@102='REC16',@103='REC13',
          @104='REC19', @105='REC14', @106='REC16')

REC53     ('TYPTABL', FNID=5, FNARG='REC65', DORC="D",
          XYLAST=106,@101='REC12',@102='REC18',@103='REC13',
          @104='REC20', @105='REC14', @106='REC20')
```

FIGURE 7 - IDS-2 REPPESENTATION OF BANK PROBLEM

```
REC54     ('TYPTABL', FNID=6, FNARG='REC65', DORC="D",
          XYLAST=106, @101='REC12', @102='REC67',
          @103='REC13', @104='REC68', @105='REC14',
          @106='REC69')

REC61     ('UNIT', NUM=1)

REC62     ('DECIMAL', NUM=333)

REC63     ('DECIMAL', NUM=583)

REC64     ('DECIMAL', NUM=1000)

REC65     ('PARAMNO', NUM=1)

REC66     ('RANDM')

REC67     ('MINUTE', NUM=50)

REC68     ('MINUTE', NUM=40)

REC69     ('MINUTE', NUM=20)

REC70     ('UNIT', NUM=2)

REC71     ('PARAMNO', NUM=2)

REC72     ('MINUTE', NUM=10)

REC73     ('MINUTE', NUM=7200)

REC74     ('UNIT', NUM=100)

REC81     ('AT', LOCOBJ='REC21')

REC82     ('IN', LOCOBJ='REC21')

REC83     ('AT', LOCOBJ='REC71')

REC91     ('SELARGS', ARGA='REC71', ARGB='REC52',
          ARGC='REC53', ARGE="Q")

REC92     ('PREDLIST',LASTREC=13,@11='REC1',
          @12='REC2',@13='REC3')

REC93     ('MOBLIST',LASTREC=14,@11='REC11',@12='REC12',
          @13='REC13',@14='REC14')

REC94     ('STALIST',LASTREC=17,@11='REC15',@12='REC16',
          @13='REC17',@14='REC18',@15='REC19',@16='REC20',
          @17='REC21')

REC95     ('ACTNLIST',LASTREC=14,@11='REC1',@12='REC2',
          @13='REC3',@14='REC4')

REC96     ('SCSRLIST',LASTREC=13,@11='REC2',@12='REC3',
          @13='REC4')
```

FIGURE 7 (CONTINUED)

```
REC97       ('DSTRLIST',LASTREC=18,@11='REC42',@12='REC43',
            @13='REC44',@14='REC45',@15='REC51',@16='REC52',
            @17='REC53',@18='REC54')


SETMEM      (RNNO(MEM)=1,MEPTR(MEM)='REC93',
            DISTPTR(MEM)='REC97',SUCPTR(MEM)='REC96',
            PROBTIME(MEM)='REC73',MEMID(MEM)=7,
            SEPTR(MEM)='REC94',ACPTR(MEM)='REC95',
            MVARID(MEM)=1,EXPUSED(MEM))
```

FIGURE 7 (CONTINUED)

```
        SIMULATE
        RMULT       277,423,715,121,655,531,999,813
WNDW2   EQU         6,F,Q
WNDW3   EQU         7,F,Q
WNDW4   EQU         8,F,Q
WNDW5   EQU         9,F,Q
WNDW6   EQU         10,F,Q
BANK    EQU         11,S,Q
11      STORAGE     100
CUST2   EQU         2,T
2       TABLE       M1,0,1,2
CUST3   EQU         3,T
3       TABLE       M1,0,1,2
CUST4   EQU         4,T
4       TABLE       M1,0,1,2
1       FUNCTION    RN1,C24
0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915/.7,1.2/.75,1.39/
.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52/.94,2.81/.95,2.99/.96,3.2/
.97,3.5/.98,3.9/.99,4.6/.995,5.3/.998,6.2/.999,7/.9997,8/
3       FUNCTION    RN2,D3
.333,CUST2/.583,CUST3/1.000,CUST4/
4       FUNCTION    P1,D3
CUST2,WNDW2/CUST3,WNDW5/CUST4,WNDW2/
5       FUNCTION    P1,D3
CUST2,WNDW4/CUST3,WNDW6/CUST4,WNDW6/
6       FUNCTION    P1,D3
CUST2,ACT3/CUST3,ACT4/CUST4,ACT5/
        GENERATE    10,FN1
        ASSIGN      1,FN3
        ENTER       BANK
        SELECTMIN   2,FN4,FN5,,Q
        TRANSFER    ,FN6
ACT3    QUEUE       P2
        SEIZE       P2
        DEPART      P2
        ADVANCE     50,FN1
        RELEASE     P2
        TRANSFER    ,ACT6
ACT4    QUEUE       P2
        SEIZE       P2
        DEPART      P2
        ADVANCE     40,FN1
        RELEASE     P2
```

FIGURE 8 - GPSS-1 REPRESENTATION OF BANK PROBLEM

```
        TRANSFER    ,ACT6
ACT5    QUEUE       P2
        SEIZE       P2
        DEPART      P2
        ADVANCE     20,FN1
        RELEASE     P2
ACT6    TABULATE    P1
        LEAVE       BANK
        TERMINATE
        GENERATE    7200
        TERMINATE   1
        START       1
        END
```

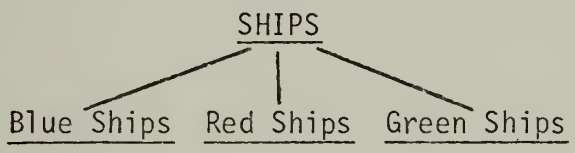
FIGURE 8 (CONTINUED)

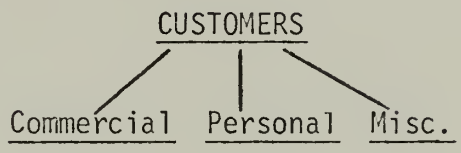
```
        SIMULATE
        RMULT       277,423,715,121,655,531,999,813
WNDW2   EQU         6,F,Q
WNDW3   EQU         7,F,Q
WNDW4   EQU         8,F,Q
WNDW5   EQU         9,F,Q
WNDW6   EQU         10,F,Q
BANK    EQU         11,S,Q
11      STORAGE     100
CUST2   EQU         2,T
2       TABLE       M1,5,3,30
CUST3   EQU         3,T
3       TABLE       M1,5,3,30
CUST4   EQU         4,T
4       TABLE       M1,5,3,30
1       FUNCTION    RN1,C24
0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915/.7,1.2/.75,1.39/
.8,1.6/.84,1.83/.88,2.12/.9,2.3/.92,2.52/.94,2.81/.95,2.99/.96,3.2/
.97,3.5/.98,3.9/.99,4.6/.995,5.3/.998,6.2/.999,7/.9997,8/
3       FUNCTION    RN2,D3
.333,CUST2/.583,CUST3/1.000,CUST4/
4       FUNCTION    P1,D3
CUST2,WNDW2/CUST3,WNDW5/CUST4,WNDW2/
5       FUNCTION    P1,D3
CUST2,WNDW4/CUST3,WNDW6/CUST4,WNDW6/
6       FUNCTION    P1,D3
CUST2,50/CUST3,40/CUST4,20/
        GENERATE    10,FN1
        ASSIGN      1,FN3
        ENTER       BANK
        SELECTMIN   2,FN4,FN5,,Q
ACT3    QUEUE       P2
        SEIZE       P2
        DEPART      P2
        ADVANCE     FN6,FN1
        RELEASE     P2
ACT4    TABULATE    P1
        LEAVE       BANK
        TERMINATE
        GENERATE    7200
        TERMINATE   1
        START       1
        END
```

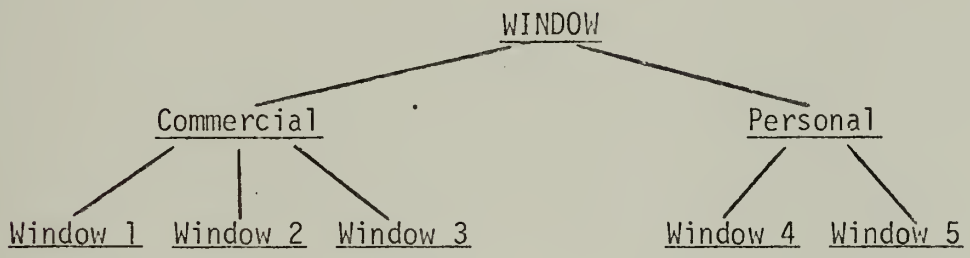
FIGURE 9 - GPSS-2 REPRESENTATION OF BANK PROBLEM

D. OTHER MODIFICATIONS

An interesting feature of the GES system is the ability to handle sub-structured entities. The structuring is demonstrated below for the harbor and bank problems of Figures 1 and 5.







The lower level entities, e.g. Green Ships, possess all the attributes associated with the entity in the level above, e.g. SHIPS. Thus, those attributes which are common to all the entities in the structure may be specified in the upper level, and only those attributes which differentiate entities must be specified at the lower level. While GES has the ability to detect an entity which is at the sub-structure level, it has no ability to detect those entities which head a structure. This ability was incorporated into XGES and was used to avoid outputting unnecessary EQU statements and table definition cards.

During the revision of the GES, the encoding was reorganized in order to increase the visibility of encoding rule processing logic. The resulting XGES encoding rules are listed in Appendix B and the attributes and named records are shown in Appendix A. Figure 10 is a final example of the simple queuing problems which XGES is capable of encoding. Figure

11 is the direct specification of the problem and Figure 12 is the resulting GPSS program.

EXAMPLE PROBLEM

Cars arrive at a gas station randomly. On the average, one car arrives every five minutes. This gas station has one pump at which cars are serviced individually. The mean service time is four minutes. The distribution of service times is uniform, with times ranging from two to six minutes.

When a car arrives at the gas station, if there are no other cars in line, the car goes to the pump, is serviced, and then leaves the gas station. If there are one, two, or three cars in the line, the car gets in line to wait for its turn at the pump. When its turn comes, it is serviced, and then leaves the gas station. If there are four cars in line, the arriving car leaves the gas station immediately.

Figure 10 - Description of Gas Station Problem

```
'REC1'   ('ARRIV',IDNO=1,LOCATION='REC21',AGENT='REC11',
          IETM='REC51',SUCC='REC71')

'REC2'   ('SERVIC',IDNO=2,PRED='REC1',SUCC='REC3',LOCATION=
           'REC21',DURATION='REC61',GOAL='REC11')

'REC3'   ('LEAV',IDNO=3,PRED='REC2',AGENT='REC11',
          LOCATION='REC21')


'REC11'  ('CAR',IDNO=1,CONSUMP='REC52',IDNAME="CAR")

'REC12'  ('GASSTA',IDNO=2,QUANTITY=1,CAPACITY='REC52',
          IDNAME="GASTA")

'REC21'  ('AT',LOCOBJ='REC12')


'REC41'  ('MOBLIST',LASTREC=11,@11='REC11')

'REC42'  ('STALIST',LASTREC=11,@11='REC12')

'REC43'  ('ACTNLIST',LASTREC=13,@11='REC1',@12='REC2',
         @13='REC3')

'REC44'  ('DSTRLIST',LASTREC=11,@11='REC61')

'REC45'  ('SCSRLIST',LASTREC=12,@11='REC2',@12='REC3')


'REC51'  ('MINUTE',NUM=5)

'REC52'  ('UNIT',NUM=1)

'REC53'  ('MINUTE',NUM=4)

'REC54'  ('MINUTE',NUM=720)

'REC55'  ('UNIT',NUM=4)

'REC56'  ('MINUTE',NUM=2)


'REC61'  ('UNIFORM',MEAN='REC53',RANGE='REC56')


'REC71'  ('QTYP',SUCARG='REC12',MAXQ='REC55',OPENACT='REC2',
          CLOSACT='REC3')


SETMEM     (RNNO(MEM)=1,MEPTR(MEM)='REC41',SEPTR(MEM)=
           'REC42',ACPTR(MEM)='REC43',DISTPTR(MEM)=
           'REC44',SUCPTR(MEM)='REC45',PROBTIME(MEM)=
           'REC54',MFNID(MEM)=3,MVARID(MEM)=1)
```

FIGURE 11 - IDS REPRESENTATION OF GAS STATION PROBLEM

```
        SIMULATE
        RMULT       277,423,715,121,655,531,999,813
GASTA   EQU         2,F,Q
CAR     EQU         1,T
1       TABLE       M1,0,1,2
        GENERATE    5
        ASSIGN      1,1
        TEST L      Q$GASTA,4,ACT3
ACT2    QUEUE       GASTA
        SEIZE       GASTA
        DEPART      GASTA
        ADVANCE     4,2
        RELEASE     GASTA
ACT3    TABULATE    P1
        TERMINATE
        GENERATE    720
        TERMINATE   1
        START       1
        END
```

FIGURE 12 - GPSS REPRESENTATION OF GAS STATION PROBLEM

IV. ENCODING SYSTEM FOR TRANSLATION FROM THE IDS TO ENGLISH

The decision to develop a system to translate the IDS version of a simulation problem into English was based on the idea that a user who was not familiar at all with GPSS would have no means of knowing if his original expression of the problem had been translated into GPSS correctly. Such a system would be useful regardless of how the IDS was created. Whether the IDS was specified directly or was created through a question-answer system or was the result of the decoding of natural language text, the re-translation into English should help to reveal any "misunderstandings."

A. OBJECTIVE AND INITIAL ASSUMPTIONS

The initial objective in developing an IDS-to-English system was to create a system capable of translating the IDS into English sentences appropriate for expressing simulation problems. The development of the system was to be based on the idea of maintaining the greatest flexibility and generality possible, so that once the system had been developed to the point that it produced simulation problem descriptions in reasonable English, only a little additional effort would allow variations in the expression of the same problem and, in general, lend some elegance to the English produced. The English description would be based on the ENTITY-ACTION-⟨at a LOCATION⟩ form inherent in the Internal Data Structure [8].

The first assumption made was that the system would consist of a set of encoding rules somewhat similar to those of XGES. The second assumption was that if the rules were properly stratified in their development, then the IDS-to-English encoding system would automatically possess

sufficient flexibility and generality to allow easy expansion of the system's capabilities. These assumptions were based upon the experience gained through the work with expanding the GES and through knowledge of the capabilities of the rule language.

B. RULE DEVELOPMENT

Once the decision was made to write encoding rules, it was necessary to develop the rules to obtain the information from the IDS in some logical order. That is, before any English text could be produced, the meaning had to be extracted from the IDS in proper semantic sequence for translation into English. The vehicle which provided this sequence was the action list of the MEMORY record. Through the action list all the necessary information could be accessed, and the order of the actions on the list is a reasonable order for the English expression of the problem. The successor attribute of each action record was used as an additional means of specifying simulation problem action and was also used to smooth the English expression of the problem flow from one action to the next.

The basic problem in writing the rules was that of ensuring that all the necessary information was carried from rule to rule until the actual characters were emitted. The delicate point was deciding between what information should be carried along to accommodate both present and future semantic capabilities of the system, and what information was excess and would only serve to slow the rule processing.

A previously unused capability of the encoding rules was employed to create a SEGMENT record from pieces of an IDS record. For example, it was necessary to loop through a PTYP record [8] and copy selected portions of three of the six records pointed to and combine them in the proper

order into a single SEGMENT record. This new SEGMENT record was used later on to output a complex sentence consisting of a series of actions. These actions were picked from the SEGMENT record easily since only the desired information was present and it was in the proper order.

Once the encoding rules were developed to extract the meaning from the IDS in the proper order for conversion to English sentences, they were combined with encoding rules previously constructed to actually output properly formed sentences. The resulting list of attributes, indicators and named records is shown in Appendix C and the encoding rules are listed in Appendix D. The English texts produced by the encoding rules of Appendix D for three sample problems are given in Figures 13, 14 and 15. They may be compared with the original expressions of the problems shown in Figures 1, 5 and 10.

THE SHIPS ARRIVE AT THE PORT. THE TIME BETWEEN ARRIVALS IS UNIFORMLY DISTRIBUTED, WITH A MEAN OF 30 MINUTES AND A HALF-RANGE OF 60 MINUTES. 20 PERCENT OF THE SHIPS ARE GREEN, 30 PERCENT ARE RED, AND THE REST ARE BLUE. AFTER ARRIVING AT THE PORT, THE SHIPS UNLOAD CARGO AT THE DOCK IN THE PORT. THERE ARE 3 DOCKS IN THE PORT. THE TIME FOR THE SHIPS TO UNLOAD CARGO AT THE DOCK IN THE PORT IS NORMALLY DISTRIBUTED, WITH A MEAN OF 180 MINUTES FOR THE GREEN SHIPS, 240 MINUTES FOR THE RED SHIPS, AND 300 MINUTES FOR THE BLUE SHIPS AND A STANDARD DEVIATION OF 30 MINUTES FOR THE GREEN SHIPS, 60 MINUTES FOR THE RED SHIPS, AND 90 MINUTES FOR THE BLUE SHIPS. AFTER UNLOADING CARGO AT THE DOCK IN THE PORT, THE GREEN SHIPS UNLOAD CARGO AT THE PIER IN THE PORT, THE RED SHIPS UNLOAD CARGO AT THE DEPOT IN THE PORT, AND THE BLUE SHIPS UNLOAD CARGO AT THE BARGE IN THE PORT. THERE ARE 2 PIERS IN THE PORT. THE TIME FOR THE GREEN SHIPS TO UNLOAD CARGO AT THE PIER IN THE PORT IS NORMALLY DISTRIBUTED, WITH A MEAN OF 60 MINUTES AND A STANDARD DEVIATION OF 15 MINUTES. AFTER UNLOADING CARGO AT THE PIER IN THE PORT, 40 PERCENT OF THE SHIPS LOAD CARGO AT THE DOCK IN THE PORT FOR 120 MINUTES, AND THE REST WAIT IN THE HARBOR IN THE PORT UNTIL THE BARGE IS AVAILABLE. THE CAPACITY OF THE DEPOT IS 4 SHIPS. THE TIME FOR THE RED SHIPS TO UNLOAD CARGO AT THE DEPOT IN THE PORT IS EXPONENTIALLY DISTRIBUTED, WITH A MEAN OF 60 MINUTES. AFTER UNLOADING CARGO AT THE DEPOT IN THE PORT, 40 PERCENT OF THE SHIPS LOAD CARGO AT THE DOCK IN THE PORT FOR 120 MINUTES, AND THE REST WAIT IN THE HARBOR IN THE PORT UNTIL THE BARGE IS AVAILABLE. THE TIME FOR THE BLUE SHIPS TO UNLOAD CARGO AT THE BARGE IN THE PORT IS EXPONENTIALLY DISTRIBUTED, WITH A MEAN OF 90 MINUTES. AFTER UNLOADING CARGO AT THE BARGE IN THE PORT, 40 PERCENT OF THE SHIPS LOAD CARGO AT THE DOCK IN THE PORT FOR 120 MINUTES, AND THE REST WAIT IN THE HARBOR IN THE PORT UNTIL THE BARGE IS AVAILABLE. AFTER LOADING CARGO AT THE DOCK IN THE PORT, THE SHIPS WAIT IN THE HARBOR IN THE PORT UNTIL THE BARGE IS AVAILABLE. AFTER WAITING IN THE HARBOR IN THE PORT, THE SHIPS LEAVE THE PORT. THE SIMULATION IS TO BE RUN FOR 5760 MINUTES.

FIGURE 13 - ENCODED ENGLISH TEXT OF HARBOR PROBLEM

THE CUSTOMERS ARRIVE AT THE BANK. THE TIME BETWEEN ARRIVALS IS EXPONENTIALLY DISTRIBUTED, WITH A MEAN OF 10 MINUTES. THE CAPACITY OF THE BANK IS 100 CUSTOMERS. 33 PERCENT OF THE CUSTOMERS ARE COMMERCIAL CUSTOMERS, 25 PERCENT ARE PERSONAL CUSTOMERS, AND THE REST ARE MISCELLANEOUS CUSTOMERS. AFTER ARRIVING AT THE BANK, THE CUSTOMERS GO TO AN APPROPRIATE WINDOW WITH THE SHORTEST LINE. THERE ARE 5 WINDOWS IN THE BANK. AFTER GOING TO AN APPROPRIATE WINDOW WITH THE SHORTEST LINE, THE COMMERCIAL CUSTOMERS ARE SERVICED THERE, THE PERSONAL CUSTOMERS ARE SERVICED THERE, AND THE MISCELLANEOUS CUSTOMERS ARE SERVICED THERE. THE TIME FOR THE COMMERCIAL CUSTOMERS TO BE SERVICED THERE IS EXPONENTIALLY DISTRIBUTED, WITH A MEAN OF 50 MINUTES. AFTER BEING SERVICED THERE, THE CUSTOMERS LEAVE THE BANK. THE TIME FOR THE PERSONAL CUSTOMERS TO BE SERVICED THERE IS EXPONENTIALLY DISTRIBUTED, WITH A MEAN OF 40 MINUTES. AFTER BEING SERVICED THERE, THE CUSTOMERS LEAVE THE BANK. THE TIME FOR THE MISCELLANEOUS CUSTOMERS TO BE SERVICED THERE IS EXPONENTIALLY DISTRIBUTED, WITH A MEAN OF 20 MINUTES. AFTER BEING SERVICED THERE, THE CUSTOMERS LEAVE THE BANK. THE SIMULATION IS TO BE RUN FOR 7200 MINUTES.

FIGURE 14 - ENCODED ENGLISH TEXT FOR BANK PROBLEM

THE CARS ARRIVE AT THE GAS STATION EVERY 5 MINUTES. AFTER ARRIVING AT THE GAS STATION, IF THE LENGTH OF THE LINE AT THE GAS STATION IS LESS THAN 4, THE CAR WILL BE SERVICED AT THE GAS STATION. OTHERWISE, THE CAR WILL LEAVE THE GAS STATION. THE TIME FOR THE CARS TO BE SERVICED AT THE GAS STATION IS UNIFORMLY DISTRIBUTED, WITH A MEAN OF 4 MINUTES AND A HALF-RANGE OF 2 MINUTES. AFTER BEING SERVICED AT THE GAS STATION, THE CARS LEAVE THE GAS STATION. THE SIMULATION IS TO BE RUN FOR 720 MINUTES.

FIGURE 15 - ENCODED ENGLISH TEXT FOR GAS STATION PROBLEM

V. CONCLUSIONS

The results discussed in the previous sections are indicative of the powerful translational capabilities inherent in NLP. An especially attractive feature of translation from source language to IDS to target language is that only the encoding rules need be changed if the desired target language is changed. The encoding rules for conversion to GPSS and English are not dependent on how the information is decoded from the source language into the IDS.

Although the IDS and the encoding rules for translation to GPSS and English are still expanding as more complicated queuing problems are being processed, the basic IDS and rule structure have remained fixed. At present some knowledge of how NLP works is still necessary in order to write encoding rules. However, with a fixed IDS and rule structure, an expanded BNF for encoding rules should be sufficient to allow further development of either of the present encoding rule systems or creation of a completely new system without knowledge of how the rules are actually processed by the FORTRAN program.

Finally, it is recommended that both XGES and the IDS-to-English encoding system be coupled with the Question-Answer system for simulation problems developed by LCDR E. S. Baker [10] to produce the prototype of a system which could be of great practical value.

APPENDIX A

XGES NUMBERED ATTRIBUTES,
INDICATORS, AND NAMED RECORDS

ATTRIBUTES:

```
SUP     1
ANMS    10
```

INDICATORS:

```
EXPUSED      1
NORMUSED     2
DOLLARSIGN   3
FLAG  4
```

NAMED RECORDS:

```
PROBTIME  ('ATTR')
RNNO  ('ATTR')
TEMP  ('ATTR')
MEPTR  ('ATTR')
SEPTR  ('ATTR')
ACPTR  ('ATTR')
DISTPTR  ('ATTR')
SUCPTR  ('ATTR')
MENID  ('ATTR')
MVARID  ('ATTR')
LISTCNTR  ('ATTR')
IDNO  ('ATTR')
LOCATION  ('ATTR')
PRED  ('ATTR')
SUCC  ('ATTR')
AGENT  ('ATTR')
GOAL  ('ATTR')
DURATION  ('ATTR')
IETM  ('ATTR')
ASNDISTR  ('ATTR')
QUANTITY  ('ATTR')
SIZE  ('ATTR')
COLOR  ('ATTR')
WEIGHT  ('ATTR')
STORIND  ('ATTR')
IDENT  ('ATTR')
STRUC  ('ATTR')
CONSUMP  ('ATTR')
CAPACITY  ('ATTR')
MEAN  ('ATTR')
RANGE  ('ATTR')
STDEV  ('ATTR')
FNID  ('ATTR')
FNARG  ('ATTR')
DORC  ('ATTR')
XYLAST  ('ATTR')
PNUM  ('ATTR')
SUCARG  ('ATTR')
MAXQ  ('ATTR')
OPENACT  ('ATTR')
CLOSACT  ('ATTR')
ARGA  ('ATTR')
ARGB  ('ATTR')
LABL  ('ATTR')
BLOKMOD  ('ATTR')
MODREL  ('ATTR')
OBJREL  ('ATTR')
IDNAME  ('ATTR')
```

```
NUM   ('ATTR')
LOCOBJ   ('ATTR')
LASTREC   ('ATTR')
CHARS   ('ATTR')
CONDENTY   ('ATTR')
ARGAZ   ('ATTR')


ARRIV   ('EVENT')
LEAV   ('EVENT')
GOTO   ('EVENT')


WAIT   ('ACTIVITY')
UNLOAD   ('ACTIVITY')
LOAD   ('ACTIVITY')
SERVIC   ('ACTIVITY')


EVENT   ('ACTION')
ACTIVITY   ('ACTION')


SHIP   ('MOBENTY')
CUSTOMER   ('MOBENTY')
CAR   ('MOBENTY')


HARBOR   ('STATENTY')
PIER   ('STATENTY')
DOCK   ('STATENTY')
PORT   ('STATENTY')
DEPOT   ('STATENTY')
BARGE   ('STATENTY')
CARGO   ('STATENTY')
BANK   ('STATENTY')
WINDOW   ('STATENTY')
GASSTA   ('STATENTY')


MOBLIST   ('RECLIST')
STALIST   ('RECLIST')
DSTRLIST   ('RECLIST')
SCSRLIST   ('RECLIST')
ACTNLIST   ('RECLIST')
PREDLIST   ('RECLIST')
SELARGS   ('RECLIST')


MOBENTY   ('ENTITY')
STATENTY   ('ENTITY')


EMPDIST   ('DISTR1')
TYPDIST   ('DISTR1')


UNIFORM   ('DISTR2')
EXPON   ('DISTR2')
NORMAL   ('DISTR2')


TYPTABL   ('TABL')


TABL   ('FNCTN')
DISTR1   ('FNCTN')


COND1   ('COND')
COND2   ('COND')
```

```
IN   ('LOCDESCR')
ON   ('LOCDESCR')
NEAR   ('LOCDESCR')
AT   ('LOCDESCR')
AROUND   ('LOCDESCR')


FRACTNL   ('SUCDSCR1')
PTYP   ('SUCDSCR1')


QTYP   ('SUCDSCR2')
STYP   ('SUCDSCR2')
FTYP   ('SUCDSCR2')


SUCDSCR1   ('SUCDESCR')
SUCDSCR2   ('SUCDESCR')


GT   ('RELIND1')
NG   ('RELIND1')
EQ   ('RELIND1')
NE   ('RELIND1')
LT   ('RELIND1')
NL   ('RELIND1')


ER   ('RELIND2')
EST   ('RELIND2')


RELIND1   ('RELTV')
RELIND2   ('RELTV')


FAST   ('RELTIME')
SLOW   ('RELTIME')


MANY   ('RELQNTY')
FEW   ('RELQNTY')


DARK   ('RELCOLR')
LIGHT1   ('RELCOLR')


HEAVY   ('RELWEIT')
LIGHT2   ('RELWEIT')


LARGE   ('RELSIZ')
SMALL   ('RELSIZ')


RELTIME   ('RELVAL')
RELQNTY   ('RELVAL')
RELCOLR   ('RELVAL')
RELWEIT   ('RELVAL')
RELSIZ   ('RELVAL')


HOUR   ('ABSTIME')
MINUTE   ('ABSTIME')
SECOND   ('ABSTIME')


TON   ('ABSWEIT')
POUND   ('ABSWEIT')
OUNCE   ('ABSWEIT')
```

```
DECIMAL  ('ABSQNTY')
UNIT  ('ABSQNTY')
PERCENT  ('ABSQNTY')


RED  ('ABSCOLR')
ORANGE  ('ABSCOLR')
YELLOW  ('ABSCOLR')
GREEN  ('ABSCOLR')
BLUE  ('ABSCOLR')
VIOLET  ('ABSCOLR')
BLACK  ('ABSCOLR')
WHITE  ('ABSCOLR')


ABSTIME  ('QUANVAL')
ABSWEIT  ('QUANVAL')
ABSQNTY  ('QUANVAL')


ABSCOLR  ('QUALVAL')


PARAMNO  ('ARGVAL')
FUNCNO  ('ARGVAL')
RANDM  ('ARGVAL')
.

RELVAL  ('VAL')
QUANVAL  ('VAL')
QUALVAL  ('VAL')
ARGVAL  ('VAL')
```

APPENDIX B

XGES ENCODING RULES

```
GPSSPROG   -->   BLOK('SIMULATE')     RMULT    ...
          SELIST(%SEPTR(MEM),LISTCNTR(MEM)=11)    ...
          MELIST(%MEPTR(MEM))    EXPFUNC     NORMFUNC    ...
          DISTLIST(%DISTPTR(MEM))     SUCLIST(%SUCPTR(MEM))   ...
          ACLIST(%ACPTR(MEM))     FINIS


SELIST(LISTCNTR(MEMORY).LT.LASTREC)    --> ...
          STENTITY(%@LISTCNTR(MEMORY)(SELIST)) ...
          SELIST(LISTCNTR(MEMORY)=LISTCNTR(MEMORY)+1)

SELIST    -->   STENTITY(%@LISTCNTR(MEM)(SELIST),
          LISTCNTR(MEM)=11)

MELIST(LISTCNTR(MEM).LT.LASTREC)    -->   MENTITY(%@LISTCNTR
          (MEM)(MELIST))  MELIST(LISTCNTR(MEM)=LISTCNTR(MEM)+1)

MELIST    -->   MENTITY(%@LISTCNTR(MEM)(MELIST),LISTCNTR(MEM)
          =11)


STENTITY(STORIND,CAPACITY)    -->   EQUCARD(%STENTITY,
          CHARS="S,Q")     BLOK('STORAGE', IDNO=IDNO(STENTITY),
          ARGB=QUANTITY($STRUC(STENTITY))*NUM(CAPACITY
          ($STRUC(STENTITY))))

STENTITY    -->   EQUCARD(%STENTITY, CHARS="F,Q")

MENTITY    -->   EQUCARD(%MENTITY,CHARS="T")    ...
           TABLECARD(%MENTITY)


EQUCARD(IDNAME,¬CLASATR)    -->  NEWLINE2   NAME(CHARS=IDNAME
          (EQUCARD))    COLUMN8    E Q U   COLUMN19   ...
          NUMBER(NUM=IDNO(EQUCARD))    ,    NAME(CHARS(EQUCARD))

EQUCARD    -->   NULL

TABLECARD(¬CLASATR)    -->   BLOK('TABLE',IDNO(TABLECARD)) ...
           NAME(CHARS="M1")    ,   NUMBER(NUM=LOWINT($STRUC
           (TABLECARD)))    ,    NUMBER(NUM=INTWIDTH($STRUC
```

```
          (TABLECARD)),NUM.EQ.0,NUM=1)   ,   NUMBER(NUM=
          NUMINT($STRUC(TABLECARD)),NUM.EQ.0,NUM=2)

TABLECARD   -->   NULL


RMULT   -->   BLOK('RMULT') ...
          OUTPUT(@13="277,423,715,121,655,531,999,813")


EXPFUNC(EXPUSED(MEM))  -->   BLOK('FUNCTION',IDNO=1)  ...
          OUTPUT(@11=1,@12=1,@13="0,0/.1,.104/.2,.222/.3,.355/") ...
          OUTPUT(@13=".4,.509/.5,.69/.6,.915/.7,1.2/.75,1.39/") ...
          OUTPUT(@11=1,@12=1,@13=".8,1.6/.84,1.83/.88,2.12/.9,") ...
          OUTPUT(@13="2.3/.92,2.52/.94,2.81/.95,2.99/.96,3.2/") ...
          OUTPUT(@11=1,@12=1,@13=".97,3.5/.98,3.9/.99,4.6/.995") ...
          OUTPUT(@13=",5.3/.998,6.2/.999,7/.9997,8/")

EXPFUNC   -->   NULL

NORMFUNC(NORMUSED(MEM))   -->   BLOK('FUNCTION',IDNO=2) ...
          OUTPUT(@11=1,@12=1,@13="0,-3/.012,-2.25/.027,-1.93/") ...
          OUTPUT(@13=".043,-1.72/.062,-1.54/.084,-1.38/") ...
          OUTPUT(@11=1,@12=1,@13=".104,-1.26/.131,-1.12/.159,") ...
          OUTPUT(@13="-1/.187,-.89/.23,-.74/.267,-.62/.334,-.43/") ...
          OUTPUT(@11=1,@12=1,@13=".432,-.17/.5,0/.568,.17/.666,") ...
          OUTPUT(@13=".43/.732,.62/.77,.74/.813,.89/.841,1/") ...
          OUTPUT(@11=1,@12=1,@13=".869,1.12/.896,1.26/.916,1.38/") ...
          OUTPUT(@13=".938,1.54/.957,1.72/.973,1.93/") ...
          OUTPUT(@11=1,@12=1,@13=".988,2.25/1,3/")

NORMFUNC   -->   NULL


DISTLIST(LISTCNTR(MEMORY).LT.LASTREC)   --> ...
          FNDEF(%@LISTCNTR(MEMORY)(DISTLIST)) ...
          DISTLIST(LISTCNTR(MEMORY)=LISTCNTR(MEMORY)+1)

DISTLIST   -->   FNDEF(%@LISTCNTR(MEMORY)(DISTLIST),
          LISTCNTR(MEM)=11)


FNDEF($'DISTR1'|$'TABL')   -->   BLOK('FUNCTION',IDNO=FNID(FNDEF),
          ARGA=FNARG(FNDEF),TEMP=XYLAST(FNDEF)-100,
          ARGB=TEMP/2,DORC=DORC(FNDEF),TEMP(MEMORY)=101) ...
          NEWLINE1   FNDEF1(%FNDEF)

FNDEF   -->   NULL
```

```
SUCLIST(LISTCNTR(MEMORY).LT.LASTREC)    --> ...
        SUCREC(%@LISTCNTR(MEMORY)(SUCLIST),
        FNID(@LISTCNTR(MEMORY)(SUCLIST))=MFNID(MEMORY)) ...
        SUCLIST(LISTCNTR(MEMORY)=LISTCNTR(MEMORY)+1)

SUCLIST    -->    SUCREC(%@LISTCNTR(MEMORY)(SUCLIST),
        FNID(@LISTCNTR(MEMORY)(SUCLIST))=MFNID(MEMORY),
        LISTCNTR(MFM)=11)
        ACLIST(%ACPTR(MEMORY),LISTCNTR(MEMORY)=11)


SUCREC($'SUCDSCR1')    -->    BLOK('FUNCTION',IDNO=MFNID(MEMORY),
        MFNID(MEMORY)=MFNID(MEMORY)+1,ARGA=SUCARG(SUCREC),
        TEMP=XYLAST(SUCREC)-100,ARGB=TEMP/2,DORC="D",
        TEMP(MEMORY)=101)    NEWLINE1    FNDEF1(%SUCREC)

SUCREC    -->    NULL


FNDEF1    -->    ARG(%@TEMP(MEMORY)(FNDEF1),TEMP(MEMORY)=TEMP
        (MEMORY)+1)    ,    ARG(%@TEMP(MEMORY)(FNDEF1)) ...
        /    FNDEF2(%FNDEF1)


FNDEF2(TEMP(MEMORY).NE.XYLAST)    --> ...
        FNDEF1(%FNDEF2,TEMP(MEMORY)=TEMP(MEMORY)+1)

FNDEF2    -->    NULL


ACLIST(LISTCNTR(MEMORY).LT.LASTREC)    --> ...
        ACT(%@LISTCNTR(MEMORY)(ACLIST)) ...
        ACLIST(LISTCNTR(MEMORY)=LISTCNTR(MEMORY)+1)

ACLIST    -->    ACT(%@LISTCNTR(MEMORY)(ACLIST),
        LISTCNTR(MEMORY)=LISTCNTR(MEMORY)+1)


ACT('ARRIV',¬PRED,ASNDISTR)    --> ...
        BLOK('GENERATE',ARGA=IETM(ACT)) ...
        BLOK('ASSIGN',ARGA=ASNDISTR(ACT)) ...
        BLOK('ENTER',ARGA=LOCOBJ(LOCATION(ACT))) ...
        SUCCBLOK(ARGA=SUCC(ACT))

ACT('ARRIV',¬PRED)    -->    BLOK('GENERATE',ARGA=IETM(ACT)) ...
        BLOK('ASSIGN',)  1  ,  NUMBER(NUM=IDNO(AGENT(ACT))) ...
```

```
          BLOK('ENTER',ARGA=LOCOBJ(LOCATION(ACT))) ...
          SUCCBLOK(ARGA=SUCC(ACT))

ACT($'ACTIVITY', STORIND($STRUC(LOCOBJ(LOCATION)))) --> ...
          BLOK('QUEUE',LABL=IDNO(ACT),ARGA=LOCOBJ(LOCATION(ACT))) ...
          BLOK('ENTER', ARGA=LOCOBJ(LOCATION(ACT)),
               ARGB=CONSUMP($STRUC(AGENT(ACT)))) ...
          BLOK('DEPART',ARGA=LOCOBJ(LOCATION(ACT))) ...
          BLOK('ADVANCE',ARGA=DURATION(ACT)) ...
          BLOK('LEAVE', ARGA=LOCOBJ(LOCATION(ACT)),
               ARGB=CONSUMP($STRUC(AGENT(ACT)))) ...
          SUCCBLOK(ARGA=SUCC(ACT))

ACT($'ACTIVITY')   -->   BLOK('QUEUE',LABL=IDNO(ACT),
          ARGA=LOCOBJ(LOCATION(ACT))) ...
          BLOK('SEIZE',ARGA=LOCOBJ(LOCATION(ACT))) ...
          BLOK('DEPART',ARGA=LOCOBJ(LOCATION(ACT))) ...
          BLOK('ADVANCE',ARGA=DURATION(ACT)) ...
          BLOK('RELEASE',ARGA=LOCOBJ(LOCATION(ACT))) ...
          SUCCBLOK(ARGA=SUCC(ACT))

ACT('GOTO')  -->  BLOK('SELECT',BLOKMOD="TMIN",
                    ARGA=ARGAZ(ACT)) ...
                  SUCCBLOK(ARGA=SUCC(ACT))

ACT('LEAV',¬SUCC)   -->   BLOK('TABULATE', LABL=IDNO(ACT)) ...
          NAME(CHARS="P1") ...
          BLOK('LEAVE', ARGA=LOCOBJ(LOCATION(ACT))) ...
          BLOK('TERMINAT')


SUCCBLOK(ARGA$'QTYP')   --> ...
          BLOK('TEST',BLOKMOD="L",ARGA(SUCCBLOK)) ...
          BLOK('TRANSFER',ARGA=OPENACT(ARGA(SUCCBLOK)))

SUCCBLOK(ARGA$'FTYP')   --> ...
          BLOK('GATE',BLOKMOD="NU",ARGA(SUCCBLOK)) ...
          BLOK('TRANSFER',ARGA=OPENACT(ARGA(SUCCBLOK)))

SUCCBLOK(ARGA$'STYP')   --> ...
          BLOK('GATE',BLOKMOD="SNF",ARGA(SUCCBLOK)) ...
          BLOK('TRANSFER',ARGA=OPENACT(ARGA(SUCCBLOK)))

SUCCBLOK   -->   BLOK('TRANSFER',ARGA(SUCCBLOK))


CONDBLOK('COND1',STORIND(CONDENTY))   --> ...
          BLOK('GATE',BLOKMOD="SNF",ARGA=CONDENTY(CONDBLOK))
```

```
CONDBLOK('COND1')    --> ...
        BLOK('GATE',BLOKMOD="NU",ARGA=CONDENTY(CONDBLOK))

CONDBLOK('COND2',STORIND(CONDENTY))    --> ...
        BLOK('GATE',BLOKMOD="SF",ARGA=CONDENTY(CONDBLOK))

CONDBLOK('COND2')    --> ...
        BLOK('GATE',BLOKMOD="U",ARGA=CONDENTY(CONDBLOK))


BLOK(¬STORIND($STRUC(ARGA)), 'ENTER'|'LEAVE')    -->    NULL

BLOK('ADVANCE',ARGA$'COND')    -->    CONDBLOK(%ARGA(BLOK))

BLOK(LABL)    -->    NEWLINE2    A  C  T    NUMBER(NUM=LABL(BLOK)) ...
        COLUMN8    BLOKNAM(SUP(BLOK))    COLUMN19    BLOK1(%BLOK)

BLOK('TRANSFER',ARGA.EQ.@LISTCNTR(MEMORY)
        (ACPTR(MEMORY)))    -->    NULL

BLOK(IDNO)    -->    NEWLINE2    NUMBER(NUM=IDNO(BLOK)) ...
        COLUMN8    BLOKNAM(SUP(BLOK))    COLUMN19    BLOK1(%BLOK)

BLOK    -->    NEWLINE8    BLOKNAM(SUP(BLOK))    COLUMN19    BLOK1(%BLOK)


BLOK1('FUNCTION',IDNO.EQ.1)    --> ...
        ARG('RANDM')    ,  C  2  4

BLOK1('FUNCTION',IDNO.EQ.2)    --> ...
        ARG('RANDM')    ,  C  2  9

BLOK1('FUNCTION')    -->    ARG(%ARGA(BLOK1)) ...
        ,    NAME(CHARS=DORC(BLOK1))    NUMBER(NUM=ARGB(BLOK1))

BLOK1('STORAGE')    -->    NUMBER(NUM=ARGB(BLOK1))

BLOK1('TRANSFER')    -->    ,    ARG(%ARGA(BLOK1))

BLOK1(¬ARGA)    -->    NULL

BLOK1(BLOKMOD)    -->    COLUMN13    NAME(CHARS=BLOKMOD(BLOK1)) ...
        COLUMN19    ARGS(%ARGA(BLOK1))

BLOK1    -->    ARGS(%ARGA(BLOK1))
```

```
BLOKNAM('TERMINAT')   -->   T  E  R  M  I  N  A  T  E

BLOKNAM('FVARIABL')   -->   F  V  A  R  I  A  B  L  E

ARGS('SELARGS')  -->  NUMBER(NUM(ARGA(ARGS)))  , ...
          ARG(%ARGB(ARGS))  ,  ARG(%ARGC(ARGS))  ,  ,  ...
          NAME(CHARS=ARGE(ARGS))

ARGS('EMPDIST')   -->   ARG(%MEAN(ARGS))   ,   ARG(%ARGS)

ARGS('TYPDIST')   -->   ARG(%PNUM(ARGS))   ,   ARG(%ARGS)

ARGS('UNIFORM')   -->   ARG(%MEAN(ARGS))   ,   ARG(%RANGE(ARGS))

ARGS('EXPON')   -->   ARG(%MEAN(ARGS))   ,  F  N  1

ARGS('NORMAL')   -->   V   NUMBER(NUM=MVARID(MEMORY)) ...
          BLOK('FVARIABL',IDNO=MVARID(MEMORY),MVARID(MEMORY)=
          MVARID(MEMORY)+1)   ARG(%MEAN(ARGS))   +  ( ...
          ARG(%STDEV(ARGS))   *  F  N  2  )

ARGS('QTYP')   -->   Q   ARG(%SUCARG(ARGS),DOLLARSIGN)  , ...
          ARG(%MAXQ(ARGS))   ,   ARG(%CLOSACT(ARGS))

ARGS('STYP'|'FTYP')   -->   ARG(%SUCARG(ARGS))   , ...
          ARG(%CLOSACT(ARGS))

ARGS   -->   ARG(%ARGS)

ARG($'FNCTN'|$'SUCDSCR1')   -->   F  N   NUMBER(NUM=FNID(ARG))

ARG('EXPON')   -->   F  N  1

ARG('NORMAL')   -->   F  N  2

ARG('PARAMNO')   -->   P   NUMBER(NUM(ARG))

ARG('RANDM')   -->   R  N   NUMBER(NUM=RNNO(MEMORY))   RNCHK

ARG('FUNCNO')   -->   F  N   NUMBER(NUM(ARG))

ARG($'ACTION')   -->   A  C  T   NUMBER(NUM=IDNO(ARG))

ARG($'ENTITY',¬IDNAME)   -->   NUMBER(NUM=IDNO(ARG))

ARG($'ENTITY',DOLLARSIGN)   -->   $  NAME(CHARS=IDNAME(ARG))
```

```
ARG($'ENTITY')  -->  NAME(CHARS=IDNAME(ARG))
ARG('DECIMAL')   -->   DECNUMB(NUM(ARG))
ARG($'QUANVAL')   -->   NUMBER(NUM(ARG))

FINIS   -->   BLOK('GENERATE',ARGA=PROBTIME(MEMORY)) ...
         BLOK('TERMINAT')   COLUMN19   1   BLOK('START') ...
         COLUMN19   1   BLOK('END')

RNCHK(RNNO(MEMORY).LT.8)   -->  ...
         NULL(RNNO(MEMORY)=RNNO(MEMORY)+1)
RNCHK   -->   NULL(RNNO(MEMORY)=1)

NEWLINE1   -->   OUTPUT(@11=1,@12=1)
NEWLINE2   -->   OUTPUT(@11=1,@12=2)
NEWLINE8   -->   OUTPUT(@11=1,@12=8)
COLUMN8   -->   OUTPUT(@12=8)
COLUMN13   -->   OUTPUT(@12=13)
COLUMN19   -->   OUTPUT(@12=19)
NAME   -->   OUTPUT(@13=CHARS(NAME))
NUMBER   -->   OUTPUT(@14=NUM(NUMBER))
DECNUMB   -->   OUTPUT(@15=NUM(DECNUMB))
NULL   -->   OUTPUT
```

APPENDIX C IDS-TO-ENGLISH ATTRIBUTES, INDICATORS, AND NAMED RECORDS

ATTRIBUTES:

SUP 1

INDICATORS:

EXPUSED 1	NORMUSED 2	DOLLARSIGN 3
MARK1 4	MARK2 5	FUTURE 6
PASSIVE 7	PASTPART 8	PRESPART 9
NUMB 11-12	SING 11	PLUR 12
E 13	VFORM 8-12	INFIN 10
LY 14	PRM 15	

NAMED RECORDS:

PROBTIME ('ATTR')
RNNO ('ATTR')
TEMP ('ATTR')
TEMP2 ('ATTR')
MEPTR ('ATTR')
SEPTR ('ATTR')
ACPTR ('ATTR')
DISTPTR ('ATTR')
SUCPTR ('ATTR')
MFNID ('ATTR')
MVARID ('ATTR')
LISTCNTR ('ATTR')
IDNO ('ATTR')
LOCATION ('ATTR')
PRED ('ATTR')
SUCC ('ATTR')
AGENT ('ATTR')
GOAL ('ATTR')
DURATION ('ATTR')
IETM ('ATTR')
ASNDISTR ('ATTR')
QUANTITY ('ATTR')
SIZE ('ATTR')
COLOR ('ATTR')
WEIGHT ('ATTR')
STORIND ('ATTR')
IDENT ('ATTR')
STRUC ('ATTR')
CONSUMP ('ATTR')
CAPACITY ('ATTR')
MEAN ('ATTR')
RANGE ('ATTR')
STDEV ('ATTR')
FNID ('ATTR')
FNARG ('ATTR')
DORC ('ATTR')
XYLAST ('ATTR')
PNUM ('ATTR')
SUCARG ('ATTR')
MAXQ ('ATTR')
OPENACT ('ATTR')
CLOSACT ('ATTR')
ARGA ('ATTR')
ARGB ('ATTR')
LABL ('ATTR')
BLOKMOD ('ATTR')
MODREL ('ATTR')
OBJREL ('ATTR')
IDNAME ('ATTR')
NUM ('ATTR')
LOCOBJ ('ATTR')
LASTREC ('ATTR')

```
CHARS  ('ATTR')
CONDENTY  ('ATTR')
ARGAZ  ('ATTR')
TYPE  ('ATTR')
LC  ('ATTR')
MOBENATR  ('ATTR')
ATTRIB  ('ATTR')
STATE  ('ATTR')
CPC  ('ATTR')
PC  ('ATTR')
LENGTH  ('ATTR')
CONDITN  ('ATTR')
AGORGL  ('ATTR')
IDSNO  ('ATTR')
ATNO  ('ATTR')
ATNO2  ('ATTR')

ARRIV  ('EVENT',AGORGL ='AGENT',E)
LEAV  ('EVENT',AGORGL='AGENT',E)
GOTO  ('EVENT',AGORGL='AGENT')
WAIT  ('ACTIVITY',AGORGL='AGENT')
UNLOAD  ('ACTIVITY',AGORGL='AGENT')
LOAD  ('ACTIVITY',AGORGL='AGENT')
SERVIC  ('ACTIVITY',AGORGL='GOAL',E)

EVENT  ('ACTION')
ACTIVITY  ('ACTION')

SHIP  ('MOBENTY')
CUSTOMER  ('MOBENTY')
CAR  ('MOBENTY')

HARBOR  ('STATENTY')
PIER  ('STATENTY')
DOCK  ('STATENTY')
PORT  ('STATENTY')
DEPOT  ('STATENTY')
BARGE  ('STATENTY')
BANK  ('STATENTY')
WINDOW  ('STATENTY')
GASSTA  ('STATENTY')

MOBLIST  ('RECLIST')
STALIST  ('RECLIST')
DSTRLIST  ('RECLIST')
SCSRLIST  ('RECLIST')
ACTNLIST  ('RECLIST')
PREDLIST  ('RECLIST')
SELARGS  ('RECLIST')

CARGO  ('ENTITY')
LINE  ('ENTITY')
MOBENTY  ('ENTITY')
STATENTY  ('ENTITY')

EMPDIST  ('DISTR1')
TYPDIST  ('DISTR1')

UNIFORM  ('DISTR2',LY)
EXPON  ('DISTR2',LY)
NORMAL  ('DISTR2',LY)

TYPTABL  ('TABL')

TABL  ('FNCTN')
DISTR1  ('FNCTN')

COND1  ('COND')
COND2  ('COND')

IN  ('LOCDESCR')
ON  ('LOCDESCR')
```

```
NEAR  ('LOCDESCR')
AT  ('LOCDESCR')
AROUND  ('LOCDESCR')

FRACTNL  ('SUCDSCR1')
PTYP  ('SUCDSCR1')

QTYP  ('SUCDSCR2')
STYP  ('SUCDSCR2')
FTYP  ('SUCDSCR2')


SUCDSCR1  ('SUCDESCR')
SUCDSCR2  ('SUCDESCR')

GT  ('RELIND1')
NG  ('RELIND1')
EQ  ('RELIND1')
NE  ('RELIND1')
LT  ('RELIND1')
NL  ('RELIND1')

ER  ('RELIND2')
EST  ('RELIND2')

RELIND1  ('RELTV')
RELIND2  ('RELTV')

FAST  ('RELTIME')
SLOW  ('RELTIME')

MANY  ('RELQNTY')
FEW  ('RELQNTY')

DARK  ('RELCOLR')
LIGHT1  ('RELCOLR')

HEAVY  ('RELWEIT')
LIGHT2  ('RELWEIT')

LARGE  ('RELSIZ')
SMALL  ('RELSIZ')

COMMERCE  ('TYPEVAL')
PERSONAL  ('TYPEVAL')
MISC  ('TYPEVAL')
COMMISC  ('TYPEVAL')
PERMISC  ('TYPEVAL')

RELTIME  ('RELVAL')
RELQNTY  ('RELVAL')
RELCOLR  ('RELVAL')
RELWEIT  ('RELVAL')
RELSIZ  ('RELVAL')

HOUR  ('ABSTIME')
MINUTE  ('ABSTIME')
SECOND  ('ABSTIME')

TON  ('ABSWEIT')
POUND  ('ABSWEIT')
OUNCE  ('ABSWEIT')

DECIMAL  ('ABSQNTY')
UNIT  ('ABSQNTY')
PERCENT  ('ABSQNTY')

RED  ('ABSCOLR')
ORANGE  ('ABSCOLR')
YELLOW  ('ABSCOLR')
GREEN  ('ABSCOLR')
BLUE  ('ABSCOLR')
```

```
VIOLET   ('ABSCOLR')
BLACK    ('ABSCOLR')
WHITE    ('ABSCOLR')

BUSY     ('ABSTATE')
FULL     ('ABSTATE')
AVAILABL     ('ABSTATE')
UNAVAIL      ('ABSTATE')

ABSTIME    ('QUANVAL')
ABSWEIT    ('QUANVAL')
ABSQNTY    ('QUANVAL')

ABSCOLR    ('QUALVAL')
ABSTATE      ('QUALVAL')
TYPEVAL      ('QUALVAL')

PARAMNO    ('ARGVAL')
FUNCNO   ('ARGVAL')
RANDM    ('ARGVAL')

RELVAL   ('VAL')
QUANVAL    ('VAL')
QUALVAL    ('VAL')
ARGVAL   ('VAL')
```

APPENDIX D IDS-TO-ENGLISH ENCODING RULES

SEMOLOGY FOR ENCODING:

```
ENGLISH    -->   ...
          CLEAR(%SEPTR(MEM),LC=11) ...
          CLEAR(%MEPTR(MEM),LC=11)   ...
          SETAGL(%ACPTR(MEM),LC=11)    ...
          NEWPAR   ...
          ACTLIST(%ACPTR(MEM),LC=11)   ...
          ENDING

CLEAR(LC.LE.LASTREC)   -->   ...
          CLEAR(-MARK1(@LC),-MARK2(@LC),LC=LC+1)

CLEAR   -->   NULL

SETAGL(LC.LE.LASTREC)   -->   ...
          SET(ASET=@LC(SETAGL)(SETAGL),-MARK1(ASET))   ...
          SETAGL(LC=LC+1)

SETAGL   -->   NULL

SET(¬AGENT(ASET),¬GOAL(ASET),MBNTY(ASET),
          AGORGL(SUP(ASET)))   -->   ...
          SET(@AGORGL(SUP(ASET))(ASET)=MBNTY(ASET))

SET(¬AGENT(ASET),¬GOAL(ASET),MBNTY(ASET))   -->   ...
          SET(AGENT(ASET)=MBNTY(ASET))

SET(¬MOBENATR(ASET),AGENT(ASET)$'MOBENTY')   -->   ...
          SET(MOBENATR(ASET)='AGENT')

SET(¬MOBENATR(ASET))   -->   ...
          SET(MOBENATR(ASET)='GOAL')

SET   -->   NULL

ACTLIST(LC.LT.LASTREC)   -->   ...
          ACTN(%@LC(ACTLIST)(ACTLIST))      ACTLIST(LC=LC+1)

ACTLIST   -->   NULL
```

```
ACTN(¬MARK1)    -->    SENT(%ACTN,-PRED,-SUCC)     ACTN(MARK1)

ACTN(IETM,IETM¬$'ABSTIME')    -->
          SENT(%ACTN,ATTRIB='IETM',-PRED)     ACTN(-IETM)

ACTN(LOCATION,¬MARK1(LOCOBJ(LOCATION)))    -->    ...
          STENDESC(%LOCOBJ(LOCATION(ACTN)),
           IDSNO=LOCOBJ(LOCATION(ACTN)),MARK1(IDSNO),
           MBNTY=@MOBENATR(ACTN)(ACTN))    .    ACTN

ACTN(DURATION,DURATION¬$'COND',DURATION¬$'ABSTIME')    -->    ...
          SENT(%ACTN,ATTRIB='DURATION',-PRED)     ACTN(-DURATION)

ACTN(ASNDISTR)    -->
          ASNDESC(%ASNDISTR(ACTN))    ACTN(-ASNDISTR)

ACTN(SUCC)    -->    SUCCDESC(%SUCC(ACTN),PRED=ACTN,-IETM(PRED),
          -DURATION(PRED),$'ACTION',MARK1(SUCC(ACTN)))

ACTN    -->  NULL

ASNDESC    -->
          RECORD('CMPLX',RP(MEM)=RECORD)    ...
          RECLOOP1(%ASNDESC,LC=11,ATNO=101,ATNO2=102)

SUCCDESC($'ACTION')    -->    SENT(%SUCCDESC,-SUCC)

SUCCDESC('PTYP')    -->
          RECORD('CMPLX',PRED(SUCCDESC),RP(MEM)=RECORD) ...
          RECLOOP2(%SUCCDESC,LC=11,ATNO=102)

SUCCDESC('FRACTNL')    -->
          RECORD('CMPLX',PRED(SUCCDESC),RP(MEM)=RECORD)    ...
          RECLOOP3(%SUCCDESC,LC=11,ATNO=101,ATNO2=102)

SUCCDESC('QTYP')    -->
          RECORD('LT',OBJREL=MAXQ(SUCCDESC),RP(MEM)=RECORD) ...
          RECORD('AT',LOCOBJ=SUCARG(SUCCDESC),
           RP2(MEM)=RECORD)    ...
          RECORD('LINE',LOCATION=RP2(MEM),LENGTH=RP(MEM),
           ATTRIB='LENGTH',-RP2(MEM),RP(MEM)=RECORD)    ...
          SUCCDQFS(%SUCCDESC,ACT1=OPENACT,ACT2=CLOSACT)

SUCCDESC('FTYP'|'STYP')    -->
          RECORD(%SUCARG(SUCCDESC),RP(MEM)=RECORD,
           ATTRIB='STATE',STATE='BUSY',-LOCATION,
           SUP(SUCCDESC).EQ.'STYP',STATE='FULL')    ...
          SUCCDQFS(%SUCCDESC,ACT1=CLOSACT,ACT2=OPENACT)
```

```
SUCCDESC    -->    NULL

STENDESC(QUANTITY.GT.1)    -->    ...
          SENT(%STENDESC,ATTRIB='QUANTITY')    ...
          STENDESC(-QUANTITY,-LOCATION)

STENDESC(NUM(CAPACITY).GT.1,NUM(CAPACITY).NE.1000)    -->    ...
          SENT(%STENDESC,ATTRIB='CAPACITY',-LOCATION)    ...
          STENDESC(-CAPACITY)

STENDESC    -->    NULL

SUCCDQFS    -->    ...
          SENT(%ACT1(SUCCDQFS),MARK1(ACT1(SUCCDQFS)),-SUCC,-PRED,
           PRED(SUCCDQFS),CONDITN=RP(MEM),-RP(MEM),FUTURE,
           SING)
          SENT(%ACT2(SUCCDQFS),MARK1(ACT2(SUCCDQFS)),-SUCC,
           -PRED,CONDITN='OTHERWIS',FUTURE,SING)


RECLOOP1(ATNO2.LE.XYLAST)    -->    ...
          RECLOOP1(REC=%@ATNO2,CPC(REC)=NUM(@ATNO),
           PC(REC)=CPC(REC)-PCPC,PCPC=CPC(REC),
           ATTRIB(REC)=CLASATR(STRUC(REC)),
           @LC(SEG)(RP(MEM))=REC,LC=LC+1,ATNO=ATNO+2,
           ATNO2=ATNO2+2)

RECLOOP1    -->    RECLOOP(%RECLOOP1)

RECLOOP2(ATNO.LE.XYLAST)    -->    ...
          RECLOOP2(REC=%@ATNO,MARK1(@ATNO),-SUCC(REC),
           @LC(SEG)(RP(MEM))=REC,LC=LC+1,ATNO=ATNO+2)

RECLOOP2    -->    RECLOOP(%RECLOOP2)

RECLOOP3(ATNO2.LE.XYLAST)    -->    ...
          RECLOOP3(REC=%@ATNO2,MARK1(@ATNO2),-SUCC(REC),
           REC2=%@MOBENATR(REC)(REC),CPC(REC2)=NUM(@ATNO),
           PC(REC2)=CPC(REC2)-PCPC,PCPC=CPC(REC2),
           @MOBENATR(REC)(REC)=REC2,@LC(SEG)(RP(MEM))=REC,
           LC=LC+1,ATNO=ATNO+2,ATNO2=ATNO2+2)

RECLOOP3    -->    RECLOOP(%RECLOOP3)

RECLOOP    -->    ...
          SENT(%RP(MEM),-RP(MEM),LR=LC(RECLOOP)-1,LC=11)
```

```
  RECORD   -->   NULL

  ENDING   -->   SENT(%MEM,ATTRIB='PROBTIME')

LEXOLOGY FOR ENCODING:

  SENT(PRED)   -->   ...
          CONJ('AFTER')   PRPTCL(%PRED(SENT))   ...
          ,   SENT(-PRED)

  SENT(CONDITN.EQ.'OTHERWIS')   -->   ...
          CONJ('OTHERWIS')   ,   SENT(-CONDITN)

  SENT(CONDITN)   -->   ...
          CONJ('IF')   FINCL(%CONDITN(SENT))   ...
          ,   SENT(-CONDITN)

  SENT('CMPLX',LC.LT.LR)   -->   ...
          FINCL(%@LC(SENT)(SENT))   ,   SENT(LC=LC+1)

  SENT('CMPLX')   -->   CONJ('AND')   FINCL(%@LC(SENT)(SENT))   .

  SENT   -->   FINCL(%SENT)   .

  FINCL(ATTRIB)   -->   ATVCL(%FINCL)

  FINCL   -->   VERBPH(%FINCL)

  PRPTCL   -->   VERBPH(%PRPTCL,PRESPART)

  INFINCL   -->   VERBPH(% INFINCL,INFIN)

  ATVCL(ATTRIB.EQ.'IETM')   -->   ...
          PHRASE(CHARS=" THE TIME BETWEEN ARRIVALS IS")   ...
          VALUE(VAL=IETM(ATVCL))

  ATVCL(ATTRIB.EQ.'DURATION')   -->   ...
          PHRASE(CHARS=" THE TIME")   INFINCL(%ATVCL)   ...
          VERB('BE',SING)   VALUE(VAL=@ATTRIB(ATVCL)(ATVCL))

  ATVCL(ATTRIB.EQ.'QUANTITY')   -->   ...
          PHRASE(CHARS=" THERE ARE")   ...
          VALUE(VAL=QUANTITY(ATVCL))   ...
          NOUNPH(%ATVCL,-ATTRIB,PLUR,PRM)
```

```
ATVCL(ATTRIB.EQ.'CAPACITY')    -->    ...
        NOUNPH(%ATVCL)      VERB('BE',SING)    ...
        VALUE(VAL=CAPACITY(ATVCL))    ...
        NOUN(SUP(MBNTY(ATVCL)),PLUR)

ATVCL(ATTRIB.EQ.'PROBTIME')    -->    ...
        PHRASE(CHARS=" THE SIMULATION IS TO BE RUN FOR")    ...
        VALUE(VAL=PROBTIME(ATVCL))

ATVCL(ATTRIB.EQ.'TYPE')    -->    ...
        RECORD(SUP(ATVCL),TYPE(ATVCL),PLUR,PRM,
         RP(MEM)=RECORD)    ...
        FINCL('BE',PREDNOM=RP(MEM),-RP(MEM),SUBJECT=
         ATVCL,-@ATTRIB(ATVCL)(SUBJECT),-ATTRIB(SUBJECT))

ATVCL(@ATTRIB$'QUALVAL')    -->    ...
        FINCL('BE',PREDADJ=@ATTRIB(ATVCL)(ATVCL),SUBJECT=
         ATVCL,-@ATTRIB(ATVCL)(SUBJECT),-ATTRIB(SUBJECT))

ATVCL    -->    ...
        NOUNPH(%ATVCL)    ...
        VERB('BE',SING)    ...
        VALUE(VAL=@ATTRIB(ATVCL)(ATVCL))
RECORD    -->    NULL

ATVPH1(@ATTRIB)    -->    PREP('WITH')    ATVPH3(%ATVPH1)

ATVPH1             -->    NULL

ATVPH2(@ATTRIB)    -->    CONJ('AND')    ATVPH3(%ATVPH2)

ATVPH2             -->    NULL

ATVPH3        -->    ADJ('A')    NOUN(SUP=ATTRIB(ATVPH3))    ...
        PREP('OF')    VALUE(VAL=@ATTRIB(ATVPH3)(ATVPH3))

VALUE(SUP(VAL).EQ.'UNIT')    -->    ADJ(NUM(VAL(VALUE)))

VALUE(VAL$'QUANVAL')    -->    NOUNPH(%VAL(VALUE))

VALUE(VAL$'RELIND1')    -->    COMPPH(SUP(VAL(VALUE)))    ...
                               OBJCOMP(%OBJREL(VAL(VALUE)))

VALUE(VAL$'DISTR2')    -->    ...
        ADV(SUP(VAL(VALUE)))    VERB('DISTRIB',PASTPART) , ...
        ATVPH1(%VAL(VALUE),ATTRIB='MEAN')    ...
        ATVPH2(%VAL(VALUE),ATTRIB='RANGE')    ...
        ATVPH2(%VAL(VALUE),ATTRIB='STDEV')
```

```
VALUE(SUP(VAL).EQ.'TYPTABL')    --->  ...
        VALOOP(%VAL(VALUE),ATNO=101,ATNO2=102)

VALUE              -->   ADJ(NUM=VAL(VALUE))

COMPPH('LT')    -->   PHRASE(CHARS=" LESS THAN")

COMPPH('EQ')    -->   PHRASE(CHARS=" EQUAL TO")

COMPPH('GT')    -->   PHRASE(CHARS=" GREATER THAN")

OBJCOMP($'QUANVAL')    -->   VALUE(VAL=OBJCOMP)

OPJCOMP($'ENTITY')    -->   NOUNPH(%OBJCOMP)

VALOOP(ATNO2.LT.XYLAST)    -->   VALOOP2(%VALOOP)    ,      ...
        VALOOP(ATNO=ATNO+2,ATNO2=ATNO2+2)

VALOOP          -->   CONJ('AND')     VALOOP2(%VALOOP)

VALOOP2      -->   VALUE(VAL=@ATNO2(VALOOP2)(VALOOP2))    ...
        PREPPH('FOR',OBJPPH=@ATNO(VALOOP2)(VALOOP2),
         PLUR(OBJPPH))

VERBPH(¬SUBJECT,MOBENATR.EQ.'AGENT')    -->    ...
        VERBPH(SUBJECT=AGENT,-AGENT)

VERBPH(¬SUBJECT)    -->  ...
        VERBPH(SUBJECT=GOAL,-GOAL,PASSIVE)

VEPBPH(¬NUMB,SUBJECT$'MOBENTY')    -->    VERBPH(PLUR)

VERBPH(¬NUMB,QUANTITY(SUBJECT).GT.1)    -->    VERBPH(PLUR)

VERBPH(¬NUMB)    -->    VERBPH(SING)

VERBPH(PRESPART)    -->    VERBPH2(%VERBPH,-NUMB)

VERBPH(INFIN)    -->    ...
        PREP('FOR')    NOUNPH(%SUBJECT(VERBPH),NUMB=
         NUMB(VERBPH),$'ENTITY',¬IDSNO,IDSNO=SUBJECT(VERBPH)) ...
        PREP('TO')    VERBPH2(%VERBPH,-NUMB)

VERBPH    -->
        NOUNPH(%SUBJECT(VERBPH),NUMB=NUMB(VERBPH),
         $'ENTITY',¬IDSNO,IDSNO=SUBJECT(VERBPH))    ...
        VERBPH2(%VERBPH)
```

```
VERBPH2   -->
          VERBPH3(%VERBPH2)     VERBPH4(%VERBPH2)

VERBPH3('GOTO')    -->    VERBPH3('GO')

VERBPH3(FUTURE)    -->
          VERB('WILL')    VERBPH3(-FUTURE,-VFORM,INFIN)

VERBPH3(PASSIVE)    -->
          VERB('BE',VFORM=VFORM(VERBPH3))    ...
          VERBPH3(-PASSIVE,-VFORM,PASTPART)

VERBPH3    -->    VERB(SUP(VERBPH3),VFORM=VFORM(VERBPH3))

VERBPH4('GOTO')    -->
          PHRASE(CHARS=" TO AN APPROPRIATE")    ...
          NOUNPH(%LOCOBJ(LOCATION(VERBPH4)),-LOCATION,PRM)    ...
          PHRASE(CHARS=" WITH THE SHORTEST LINE")

VERBPH4(AGENT)    -->
          PREPPH('BY',OBJPPH=AGENT(VERBPH4))    ...
          VERBPH4(-AGENT)

VERBPH4(GOAL)    -->
          NOUNPH(%GOAL(VERBPH4))    ...
          VERBPH4(-GOAL)

VERBPH4(LOCATION,SUP(LOCOBJ(LOCATION)).EQ.'PARAMNO')    -->    ...
          PRON('THERE')     VERBPH4(-LOCATION)

VERBPH4(LOCATION,'LEAV')    -->
          NOUNPH(%LOCOBJ(LOCATION(VERBPH4)))    VERBPH4(-LOCATION)

VERBPH4(LOCATION)    -->
          PREPPH(%LOCATION(VERBPH4),OBJPPH=LOCOBJ)    ...
          VERBPH4(-LOCATION)

VERBPH4(IETM$'ABSTIME')    -->
          PREPPH('EVERY',OBJPPH=IETM(VERBPH4))

VERBPH4(DURATION$'ABSTIME')    -->
          PREPPH('FOR',OBJPPH=DURATION(VERBPH4))

VERBPH4(DURATION$'COND')    -->
          CONJ('UNTIL')    FINCL(%CONDENTY(DURATION(VERBPH4)),
           ATTRIB='STATE',STATE='AVAILABL',-LOCATION,
           SUP(DURATION(VERBPH4)).NE.'COND1',
           STATE='UNAVAIL')
```

```
VERBPH4(PREDADJ)    -->    ADJ(SUP=PREDADJ(VERBPH4))

VERBPH4(PREDNOM)    -->    NOUNPH(%PREDNOM(VERBPH4))

VERBPH4         -->    NULL

PREPPH      -->
          PREP(SUP(PREPPH))     NOUNPH(%OBJPPH(PREPPH))

NOUNPH(PC,PC.EQ.CPC)    -->
          ADJ(NUM=PC(NOUNPH)/10)     NOUN('PERCENT')     ...
          PREPPH('OF',OBJPPH=NOUNPH,-PC(OBJPPH),
           -COLOR(OBJPPH),-TYPE(OBJPPH),PLUR(OBJPPH))

NOUNPH(PC,CPC.GT.990)    -->    PHRASE(CHARS=" THE REST")

NOUNPH(PC)    -->    ADJ(NUM=PC(NOUNPH)/10)     NOUN('PERCENT')

NOUNPH($'QUANVAL')    -->
          ADJ(NUM(NOUNPH))     NOUN(%NOUNPH,PLUR,NUM.GT.1,SING)

NOUNPH(^'CARGO')    -->    NOUNPH(PRM)

NOUNPH(¬PRM)    -->    ADJ('THE')     NOUNPH(PRM)

NOUNPH(ATTRIB)    -->    NOUN(SUP=ATTRIB(NOUNPH))     ...
          PREPPH('OF',OBJPPH=NOUNPH,-@ATTRIB(NOUNPH)(OBJPPH),
           -ATTRIB(OBJPPH),-PRM(OBJPPH))

NOUNPH(COLOR)    -->
          ADJ(SUP=COLOR(NOUNPH))    NOUNPH(-COLOR)

NOUNPH(TYPE)    -->
          ADJ(SUP=TYPE(NOUNPH))  NOUNPH(-TYPE)
NOUNPH(LOCATION)    -->    NOUNPH(-LOCATION)     ...
          PREPPH(%LOCATION(NOUNPH),OBJPPH=LOCOBJ)

NOUNPH    -->    NOUN(%NOUNPH)

VERB    -->    #    VERBP(%VERB,E*)

NOUN    -->    #    NOUNP(%NOUN)

ADV    -->    #    ADVP(%ADV)

ADJ    -->    #    ADJP(%ADJ)

PREP    -->    #    PREPP(%PREP)
```

```
CONJ    -->    #    CONJP(%CONJ)
PRON    -->    #    PRONP(%PRON)

  MORPHOLOGY FOR ENCODING:

VERBP('BE',SING)    -->    I  S
VERBP('BE',PLUR)    -->    A   R    E
VERBP(PRESPART)    -->    VERBS(SUP(VERBP))    I  N  G
VERBP(PASTPART)    -->    VERBS(SUP(VERBP))    E    D
VERBP(E,SING)    -->    VERBS(SUP(VERBP))    E    S
VERBP(E,PLUR|INFIN)    -->    VERBS(SUP(VERBP))    E
VERBP(SING)    -->    VERBS(SUP(VERBP))    S

NOUNP(PLUR)    -->    NOUNS(SUP(NOUNP))    S
NOUNP    -->    NOUNS(SUP(NOUNP))

VERBS('DISTRIB')    -->    STEM(CHARS="DISTRIBUT")

NOUNS('GASSTA')    -->    NAME(CHARS="GAS STATION")
NOUNS('RANGE')    -->    NAME(CHARS="HALF-RANGE")
NOUNS('STDEV')    -->    NAME(CHARS="STANDARD DEVIATION")

ADVP(LY*)    -->    ADJP(SUP(ADVP))    L    Y

ADJP(NUM)    -->    NUMBER(%ADJP)
ADJP('EXPON')    -->    WORD(CHARS="EXPONENTIAL")
ADJP('COMMERCE')    -->    WORD(CHARS="COMMERCIAL")
ADJP('MISC')    -->    WORD(CHARS="MISCELLANEOUS")
ADJP('UNAVAIL')    -->    U  N  ADJP('AVAILABL')
ADJP('AVAILABL')    -->    WORD(CHARS="AVAILABLE")

CONJP('OTHERWIS')    -->    WORD(CHARS="OTHERWISE")

NEWPAR    -->    OUTPUT(@11=1,@12=7)
PHRASE    -->    OUTPUT(@13=CHARS(PHRASE))
WORD    -->    OUTPUT(@13=CHARS(WORD))
STEM    -->    OUTPUT(@13=CHARS(STEM))
NAME    -->    OUTPUT(@13=CHARS(NAME))
NUMBER    -->    OUTPUT(@14=NUM(NUMBER))
NULL    -->    OUTPUT
```

LIST OF REFERENCES

1. Chomsky, Noam, Aspects of the Theory of Syntax, MIT Press, Cambridge Mass., 1965.

2. Lamb, Sydney M., Outline of a Stratificational Grammar, Georgetown University Press, Washington, D. C., 1966.

3. Lamb, Sydney M., "Linguistic and Cognitive Networks," Linguistic Automation Project, Yale University, June 1969.

4. White, John, "Language and the Brain," Yale Alumni Magazine, v.33, n. 3, p. 47-51, December 1969.

5. Simmons, Robert F., "Natural Language Question-Answering Systems: 1969," Comm. ACM, v. 13, n. 2, p. 15-30, January 1970.

6. Thompson, F. B., and others, "REL: A Rapidly Extensible Language System," Proceedings of the 24th National Conference of the ACM, 1969.

7. Heidorn, George E., Natural Language Inputs to a Simulation Programming System, Research Proposal submitted to the Office of Naval Research, October 1970.

8. Hansen, Richard C., GES: A Data-Structure-to-GPSS Encoding System, M.S. Thesis, U. S. Naval Postgraduate School, December 1970.

9. International Business Machines Corporation, General Purpose Simulation System/360, User's Manual, 1968.

10. Baker, Eldon S., Question-Answer Inputs to a Simulation Program Generating System, M.S. Thesis, U. S. Naval Postgraduate School, June 1971.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Assistant Professor G. E. Heidorn, Code 55Hd Department of Operations Analysis Naval Postgraduate School Monterey, California 93940	5
4.	Assistant Professor Gordon Syms, Code 53ZZ Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
5.	LT Robert T. McGee USS Harder (SS-568) FPO San Francisco 96601	1

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY (Corporate author)

Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3 REPORT TITLE

The Translation of Data Structure Representations of Simple Queuing Problems into GPSS Programs and English Text

4 DESCRIPTIVE NOTES (Type of report and inclusive dates)

Master's Thesis; June 1971

5 AUTHOR(S) (First name, middle initial, last name)

Robert Thomas McGee

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1971	75	10

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

One of the goals of computer technology is to have the ability to communicate with the computer in a natural language such as English. A research effort underway at the Naval Postgraduate School involves the design and implementation of a computer system for translating natural language descriptions of simulation problems into executable computer programs. In this system, English text is translated into an internal data structure which is then translated into a computer program for performing the simulation.

This thesis reports on an effort made to aid the user of this system by (1) extending the capabilities of an existing procedure for translating the internal data structure into a GPSS simulation program, and (2) developing a procedure for translating the data structure into English text so the user could see that his input text had been correctly interpreted. The basic operation of the system is described and examples are given to illustrate the system's capabilities.

DD FORM 1 NOV 65 1473 (PAGE 1)

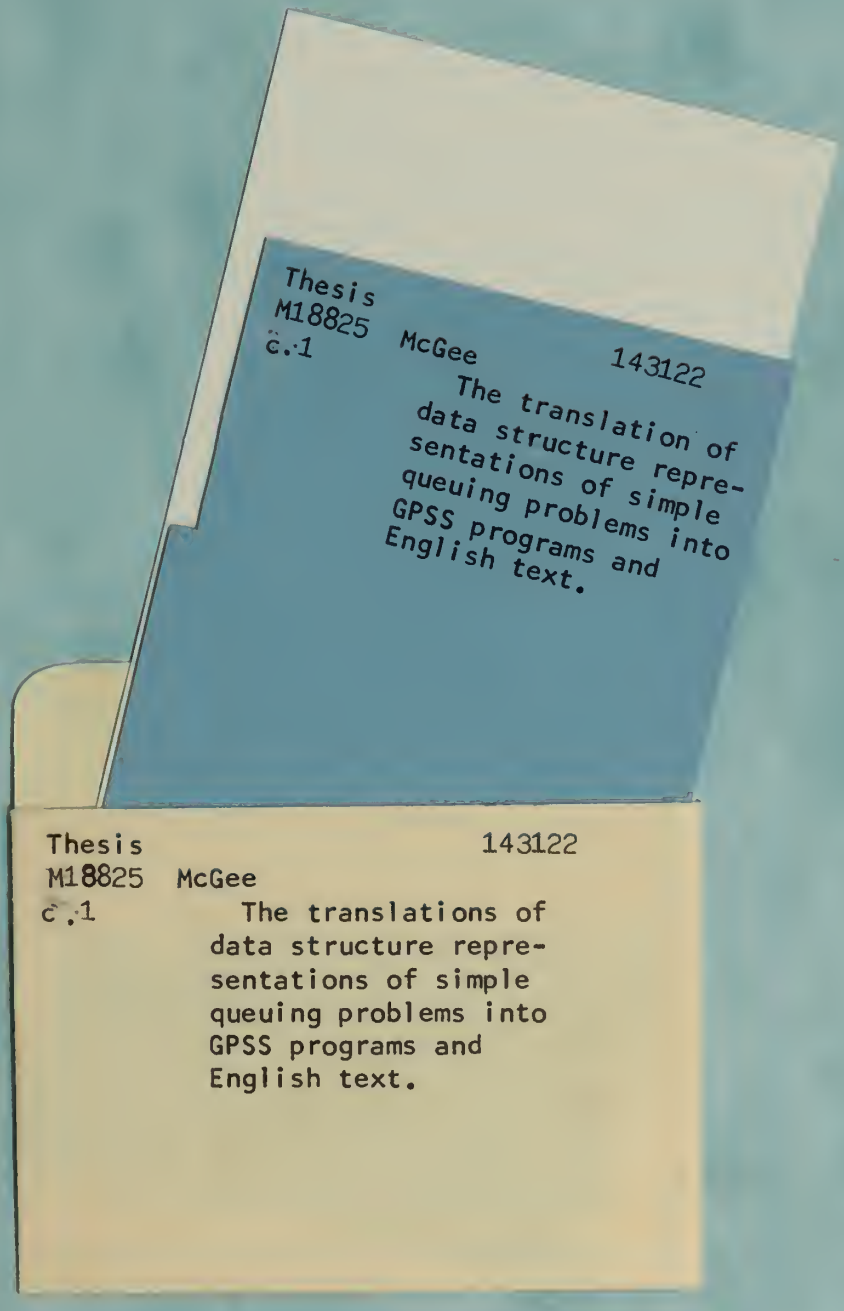
S/N 0101-807-6811

Security Classification

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Natural Language Processor						
Simulation Programming						
GPSS Programs						
Queuing Problems						

D FORM 1 NOV 65 1473 (BACK)

S/N 0101-807-6821



Thesis
M18825
c.1
McGee
143122
The translations of data structure representations of simple queuing problems into GPSS programs and English text.